Table of Contents

1

Introduction to eArm

The eArm robot is an ESP32-powered robotic arm, it offers a hands-on way to explore robotics, coding, and electronics in an engaging platform.

Features:

① **Quick Assembly & Ready-to-Use**

▸ The robotic arm is designed for effortless assembly—you can build it and start experimenting within 1–2 hours.

② **Pre-Programmed for Instant Operation**

▸ The control board comes pre-flashed with firmware, eliminating the need for initial code uploads. Simply power it on and begin using it right away.

③ **Integrated Power Management**

▸ Features an onboard battery holder with a power indicator. The robot can be charged directly via USB, ensuring uninterrupted operation.

④ **Precision Control with Joystick**

▸ The included ergonomic joystick delivers stable control, offering intuitive and responsive manipulation of the robotic arm.

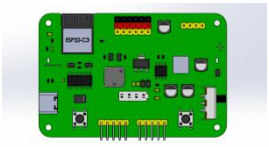
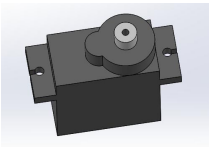
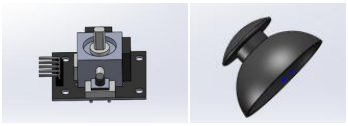
⑤ **Web-Based Control**

▸ Supports cross-platform control via a dedicated Web APP, accessible from any device with a browser—no additional installations required.

⑥ **Expandable Learning with Arduino**

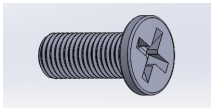
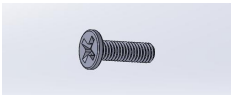
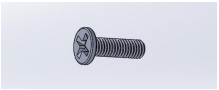
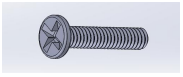
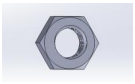
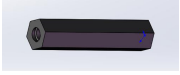
▸ Includes step-by-step Arduino programming tutorials, helping users advance from basic operations to custom robotics development.

1.1 Product List

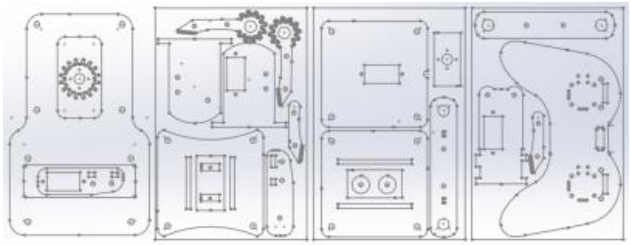
#Part 1 : Electronics Components		
Control Board 1PCS	Servo 4 PCS	5P Dupont Wire 2PCS
		
Joystick 2PCS	USB Cable 1PCS	
		

#Part 2 : Fasteners / Accessories

The following accessories listed in sequential order are all packaged in a small bag with a label and serial number attached. You can quickly locate them by referring to the serial number and name on the bag.

1 M4×10mm Screw 21Pcs	2 M3×14mm Screw 3Pcs	3 M3×10mm Screw 19Pcs
		
4 M2×10mm Screw 9Pcs	5 M1.4×5mm Screw 8Pcs	6 M4 Nut 5Pcs
		
7 M3 Lock Nut 9Pcs	8 M2 Nut 10Pcs	9 M3×7mm Standoff 5Pcs
		
10 M3×28mm Standoff 4Pcs	11 M3×40mm Standoff 2Pcs	12 M4×20mm Standoff 8Pcs
		
13 M3×6mm Screw 9Pcs	14 Flanged Bearing 4Pcs	Turntable 1PCS
		

#Part 3 : Acrylic Sheet



#Part 4 : Tools

		
M3 screwdriver 1PCS	M1.5 screwdriver 1PCS	wrench 1PCS

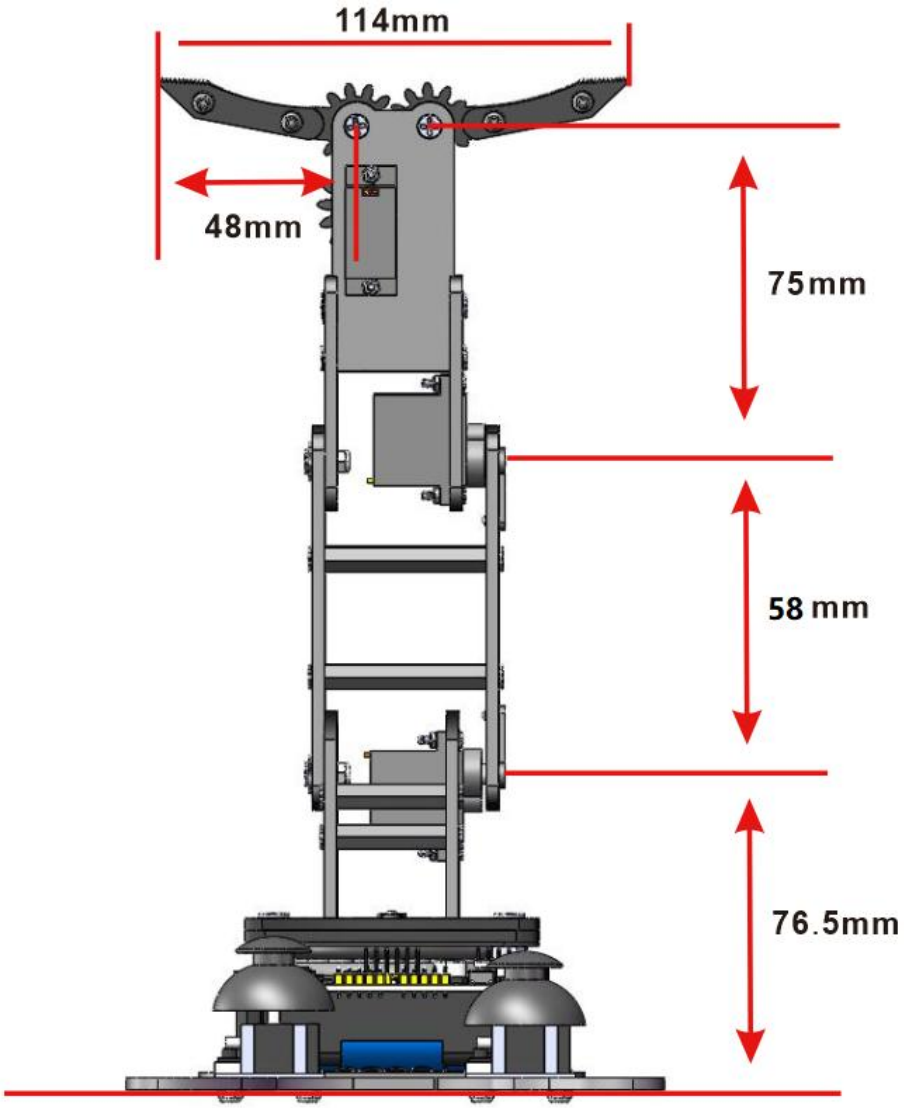
Note：You need to buy a 18650 lithium battery yourself!

Parameters of 18650 lithium battery

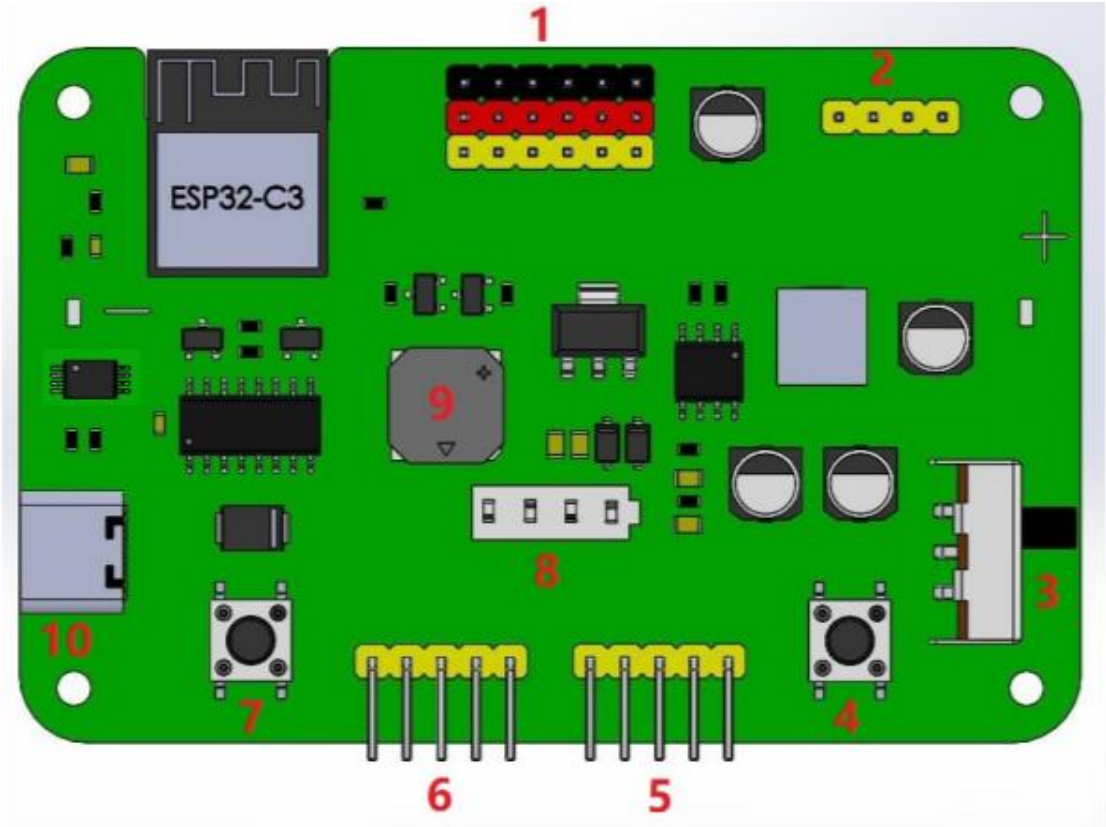
- Model: 18650 lithium battery
- Top type: Both button top and flat top are acceptable
- Capacity: >2000mAh
- Maximum charging voltage: 4.2V
- Nominal voltage: 3.7V
- End-off voltage: 2.75V
- Minimum charging current: >2A
- Minimum discharge current: >4A

1.2 Parameters of the eArm:

Control Board	eArm-ESP32-C3-MINI-1
Programming language	Arduino
Charge/discharge	5V/2A
Servo type	MG90S 180°
Remote control method	Joystick or Web APP
Assembly time required	1-2 hours
Product size	15X9.5X26MM
Packaging size	18.5X12.5X4.3MM
Applicable age	experienced players of any age or beginner aged 12+
Battery required	a 18650 battery(Not included)



1.3 Introduction to ESP32 Control Board:



■ 1: IO port, 5V/2A driving capability.

■ 2: IO port and serial port expansion port.

■ 3: Power switch.

■ 4: Battery level display button(When the power switch is off, pressing it briefly can display the battery level).

■ 5: IO port, 3.3V/500mA driving capability.

■ 6: IO port, 3.3V/500mA driving capability.

■ 7: Reset button.

■ 8: Battery level indicator LED.

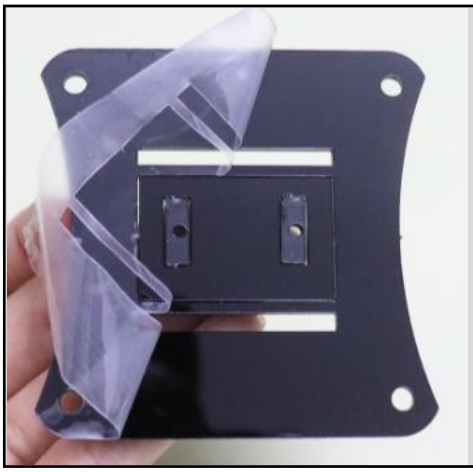
■ 9: Buzzer.

■ 10: USB Type-c Port for charging or uploading code

2

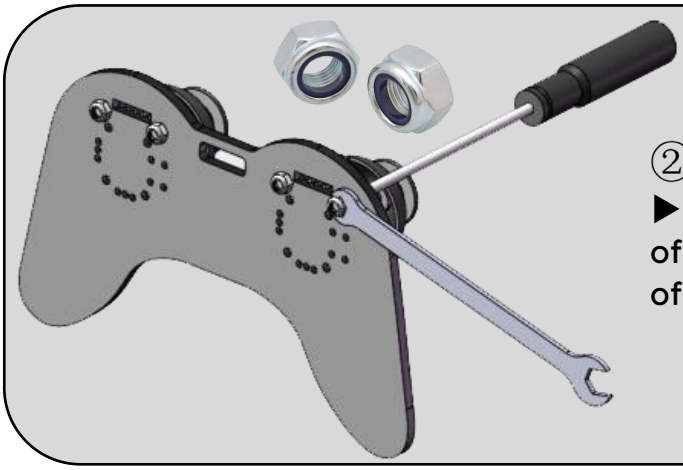
Assemble the eArm

Before You Begin: Important Notes



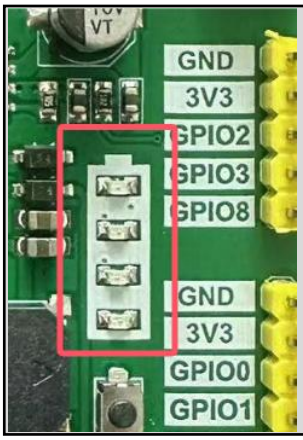
①

▶The surface of the acrylic board is covered with a protective film, please tear it off before installation!



②

▶Installation and removal of Lock nuts require the use of a hex wrench.



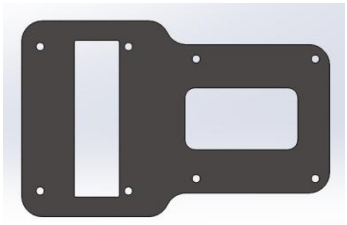
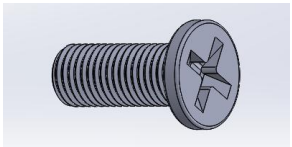
③

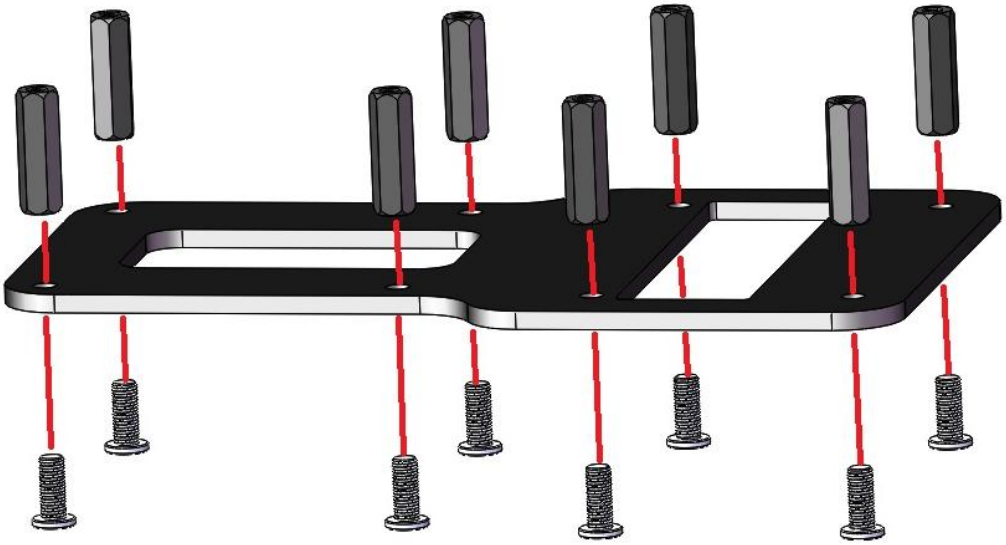
A minimum of three battery bars is required to ensure proper installation and operation of the robot. After installing an 18650 battery into the battery holder, check the power indicator. If it shows less than three bars, please charge the battery either:

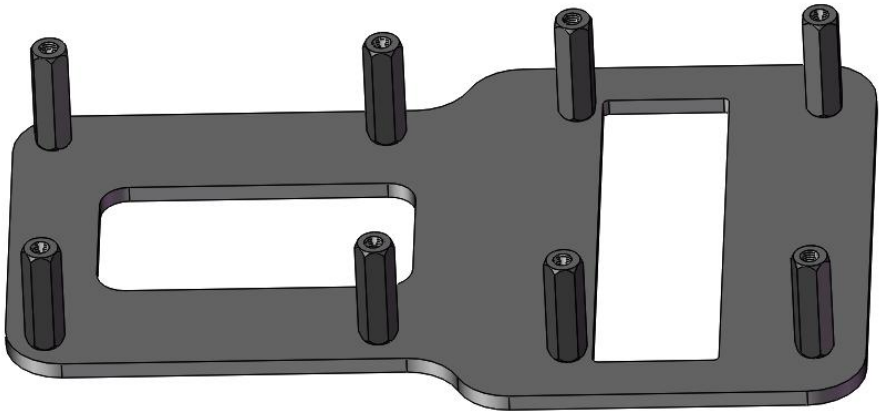
▶Using an 18650 dedicated charger

▶Via the board's charging port with a USB cable.

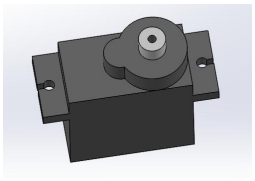
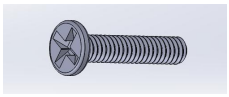
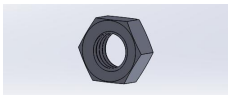
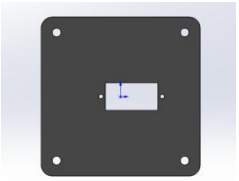
Step 1 Assemble the Robot Base

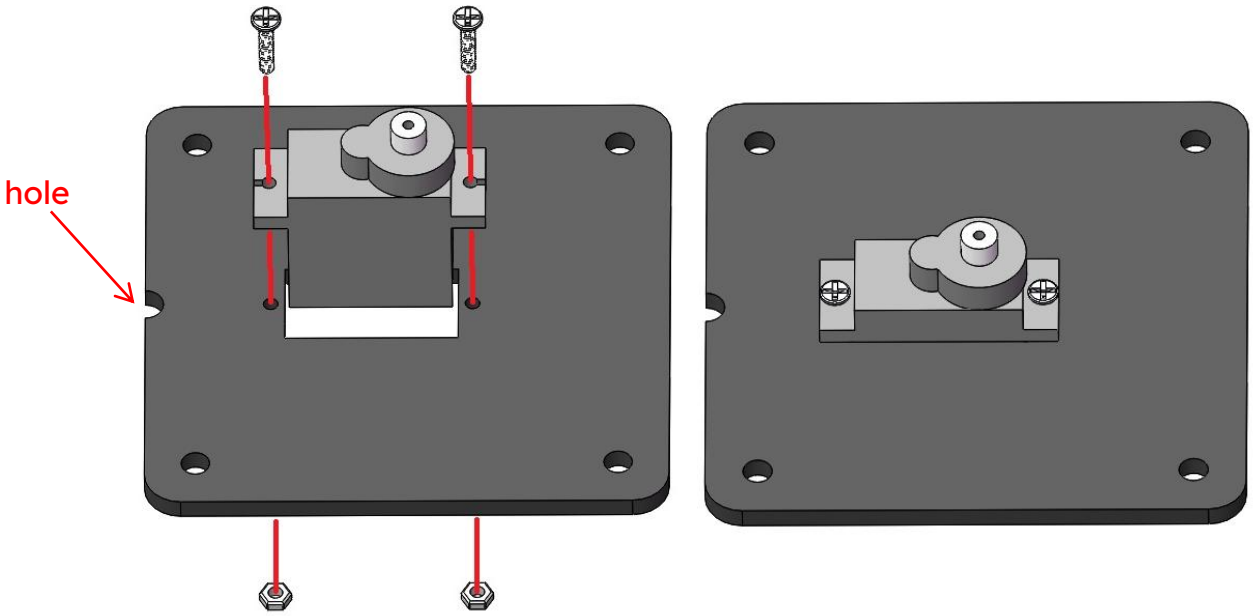
Acrylic Sheet 1PCS	**Bag No.⑫** M4X20MM Standoff 8PCS	**Bag No.①** M4X10MM Screw 8PCS
		



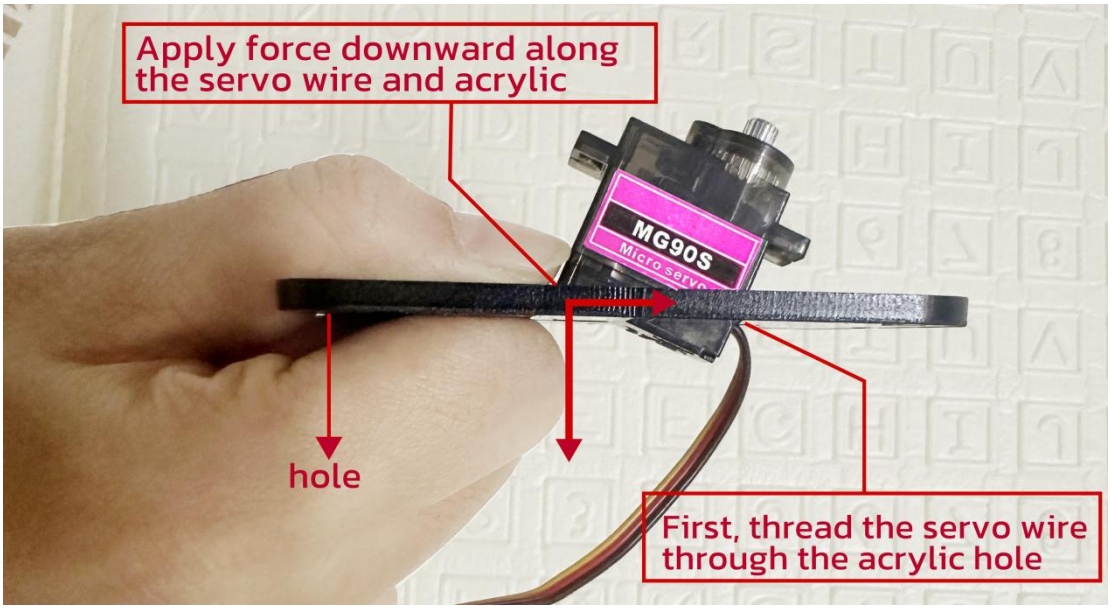


Step 2 Assemble Servo A on Robot Base

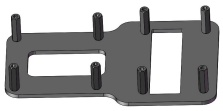
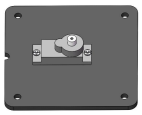
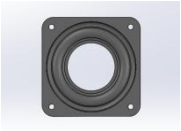
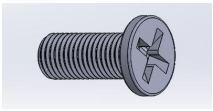
Servo 1 PCS	**Bag No.④** M2x10MM Screw 2PCS	**Bag No.⑧** M2 Nut 2PCS	Acrylic Sheet 1PCS
			

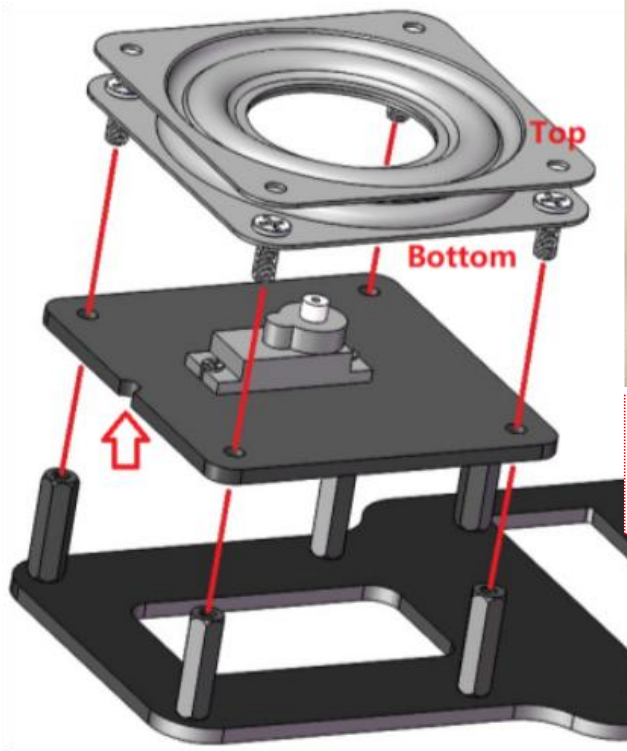


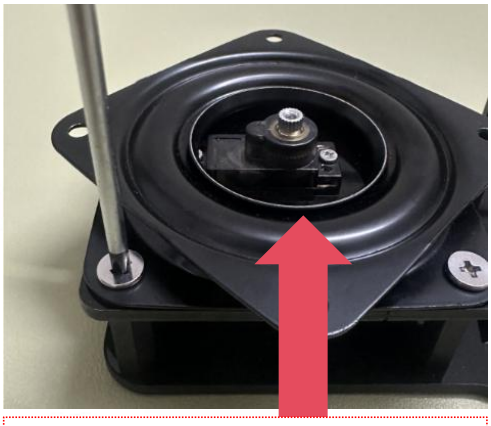
Note:



Step 3 Assemble the Turntable

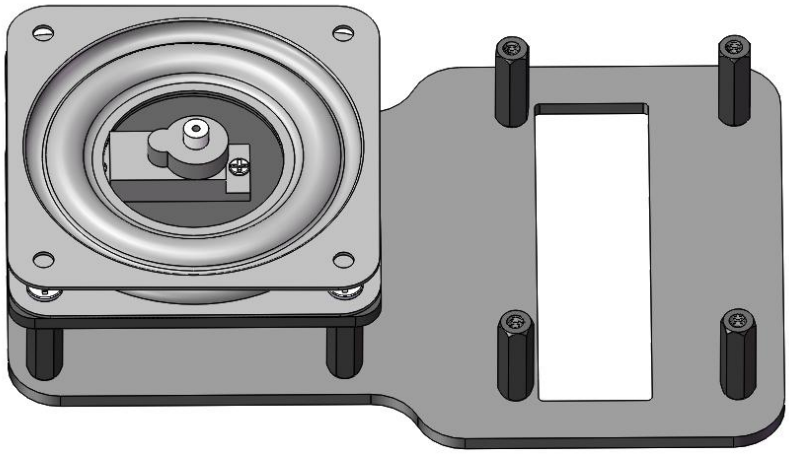
Step 1 Structure	Step 2 Structure	Turntable 1PCS	**Bag No.①** M4X10MM Screw 4PCS
			



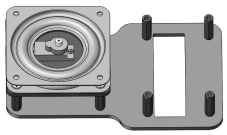


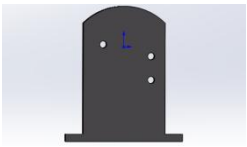
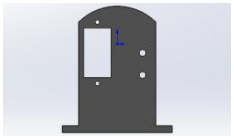
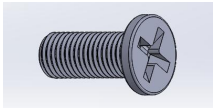
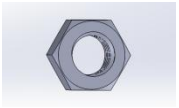
The innermost circular edge of the turntable should face upward

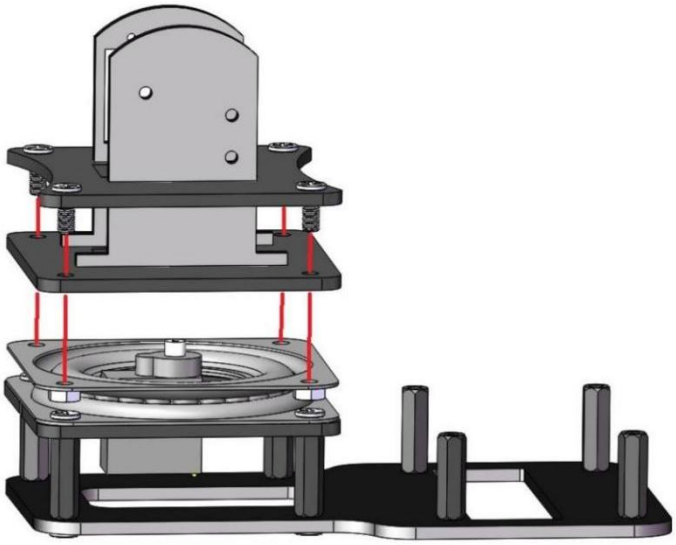
---

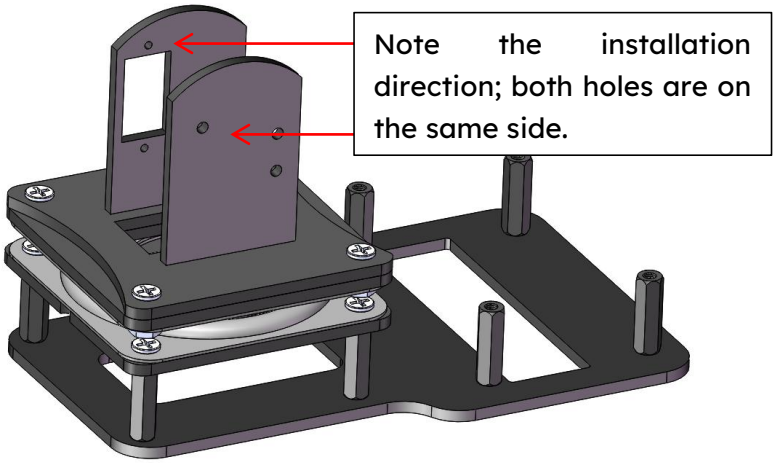


Step 4 Assemble the Lower arm

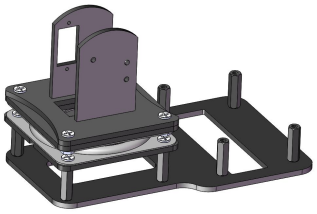
Step 3 Structure	Acrylic Sheet 1PCS	Acrylic Sheet 1PCS
		

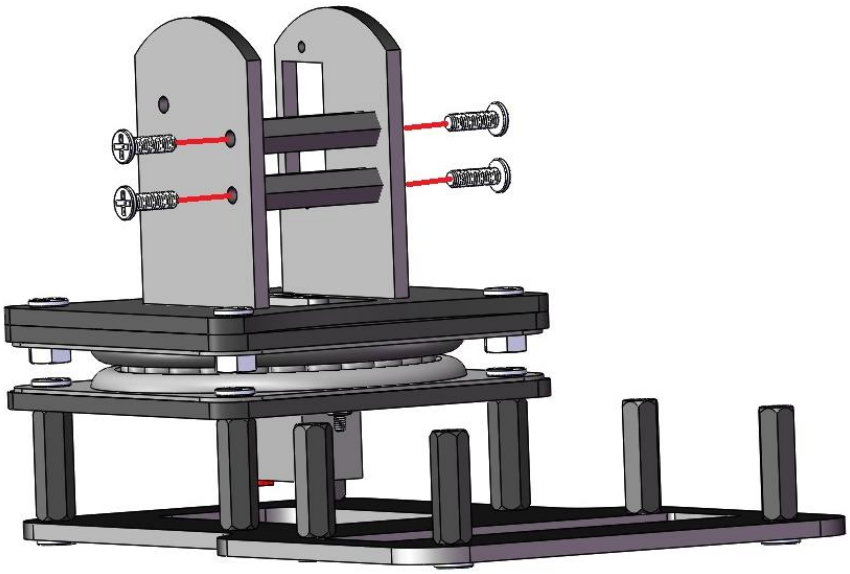
Acrylic Sheet 1PCS	Acrylic Sheet 1PCS	**Bag No.①** M4X10MM Screw 4PCS	**Bag No.⑥** M4 Nut 4PCS
			

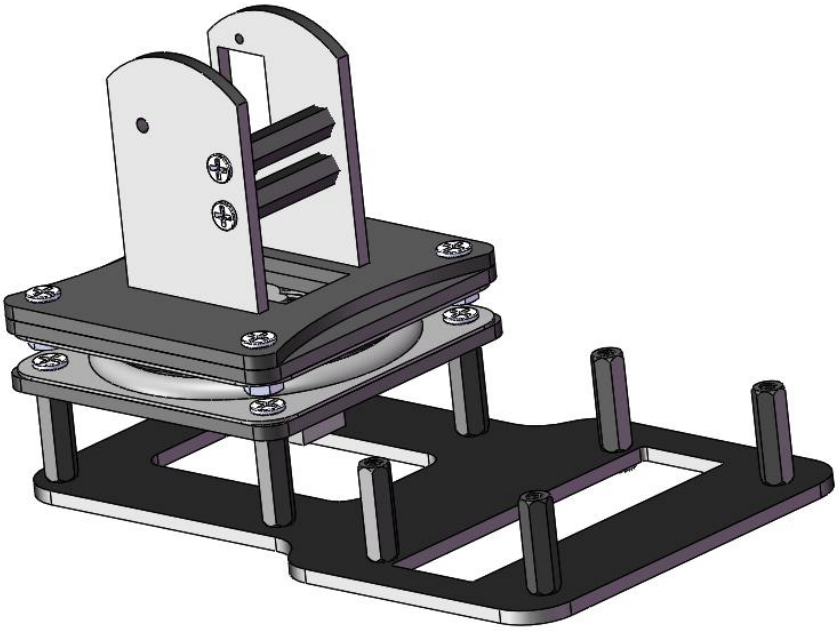




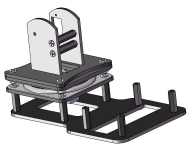
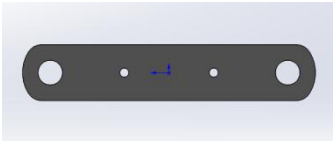
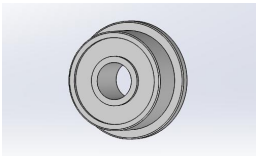
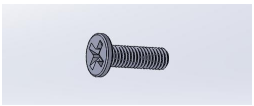
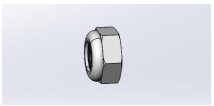
Step 5 Fix the lower arm

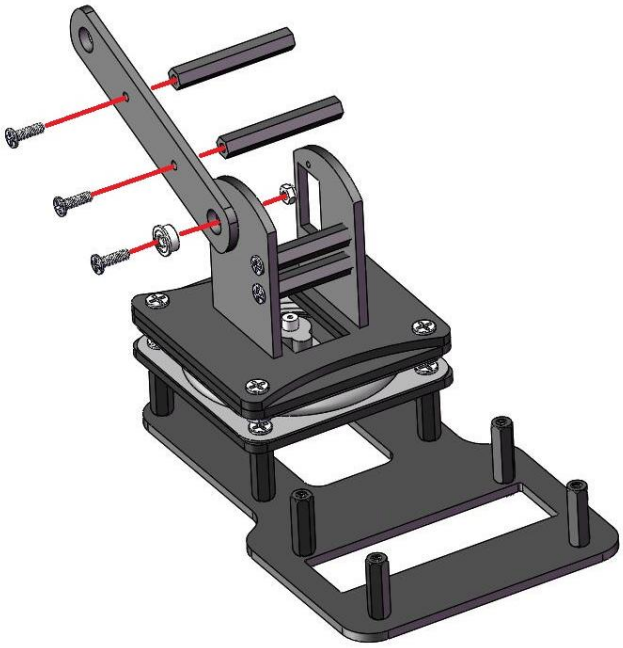
Step 4 Structure	**Bag No.⑩** M3x28MM Standoff 2PCS	**Bag No.③** M3X10MM Screw 4PCS
		



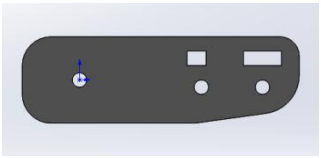
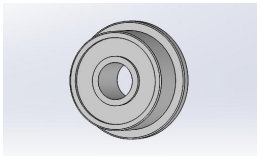
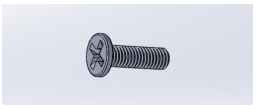


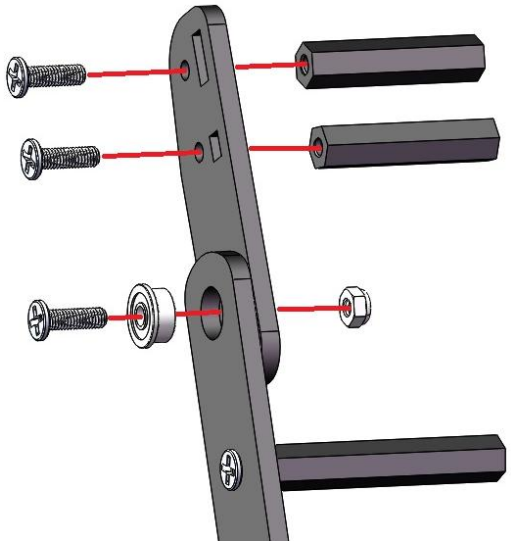
Step 6 Assemble the Middle Arm

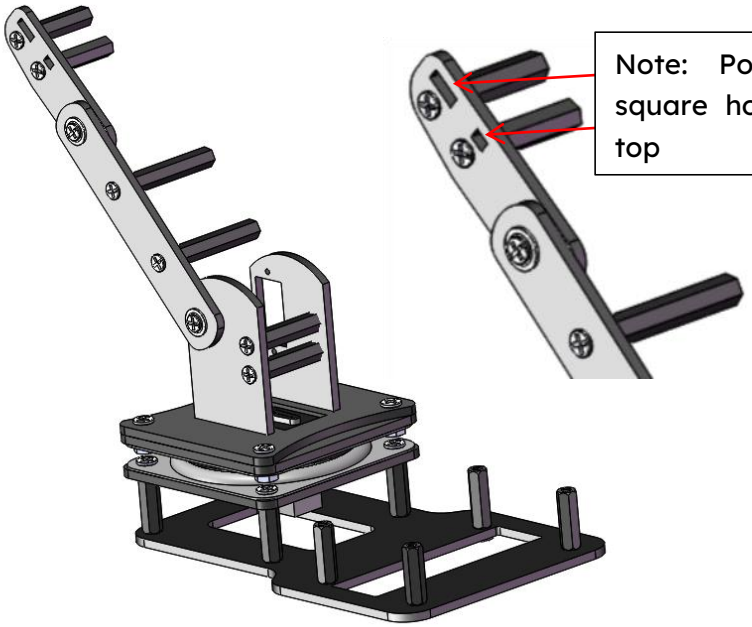
Step 5 Structure	Acrylic Sheet 1PCS	**Bag No.⑭** Flange Bearing 1PCS
		
Bag No.⑪ M3x40MM Standoff 2PCS	**Bag No.③** M3X10MM Screw 3PCS	**Bag No.⑦** M3 Lock Nut 1PCS
		



Step 7 Assemble the Upper Arm

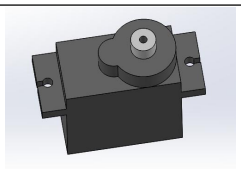
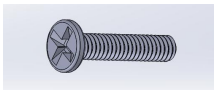
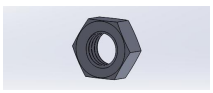
Step 6 Structure	Acrylic Sheet 1PCS	**Bag No.⑭** Flange Bearing 1PCS
		
Bag No.⑩ M3x28MM Standoff 2PCS	**Bag No.③** M3X10MM Screw 3PCS	**Bag No.⑦** M3 Lock Nut 1PCS
		

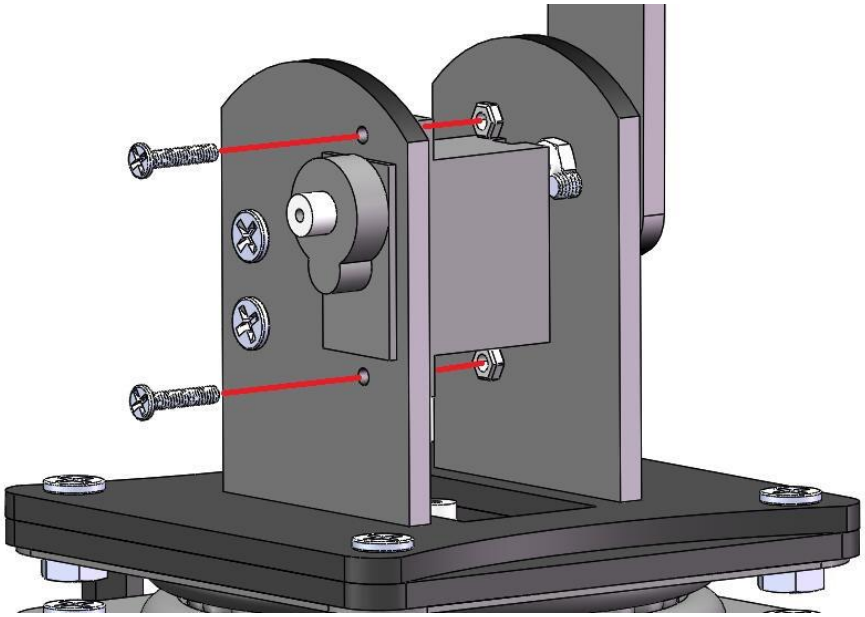


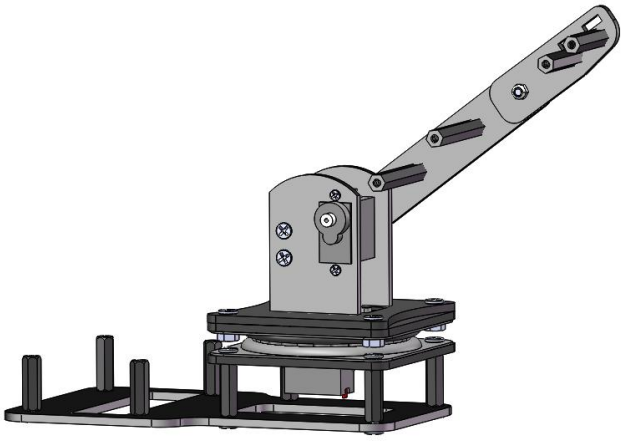


Note: Position the square holes at the top

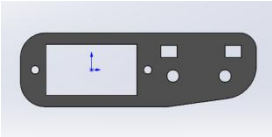
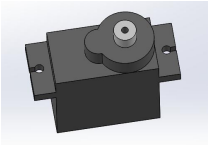
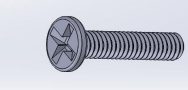
Step 8 Assemble the Servo B on lower arm

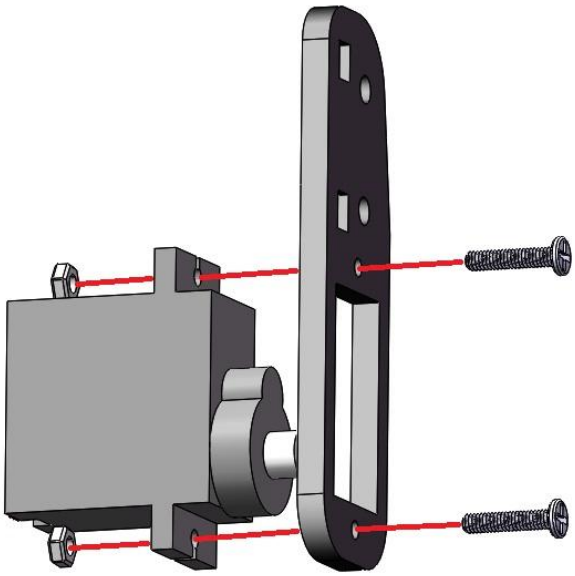
Step 7 Structure	Servo 1PCS	**Bag No.④** M2x10MM Screw 2PCS	**Bag No.⑧** M2 Nut 2PCS
			

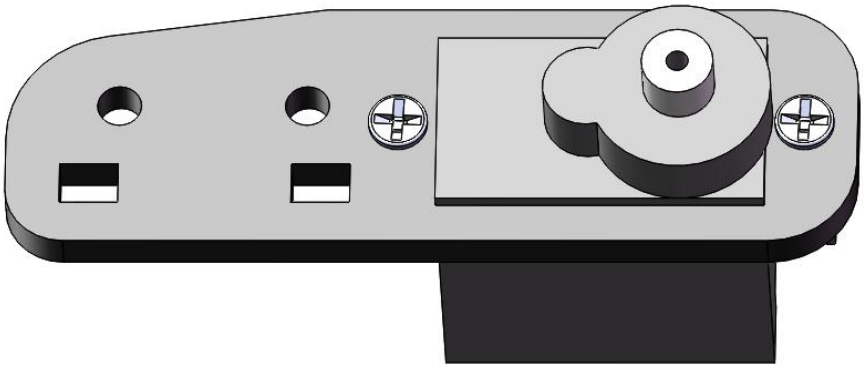




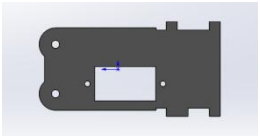
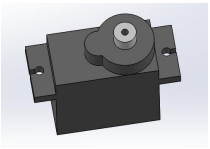
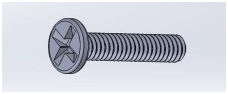
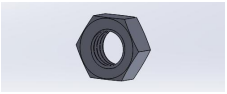
Step 9 Assemble the Servo C on Middle Arm

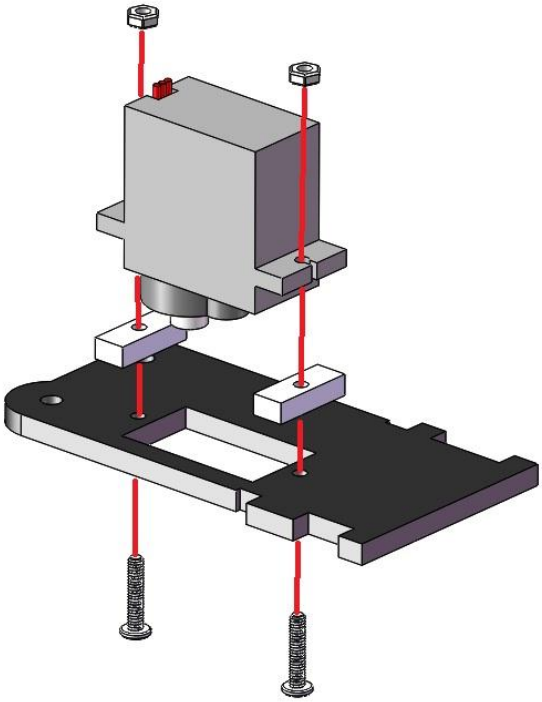
Acrylic Sheet 1PCS	Servo 1PCS	**Bag No.④** M2x10MM Screw 2PCS	**Bag No.⑧** M2 Nut 2PCS
			

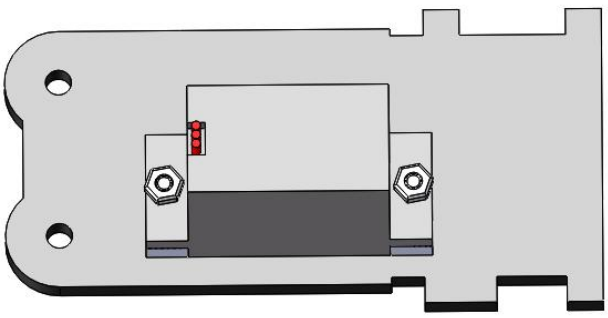




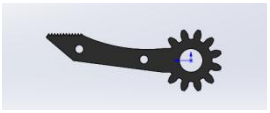
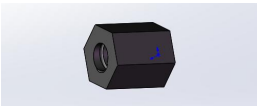
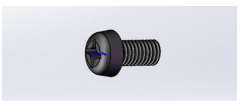
Step 10 Assemble the Servo D on Upper Arm

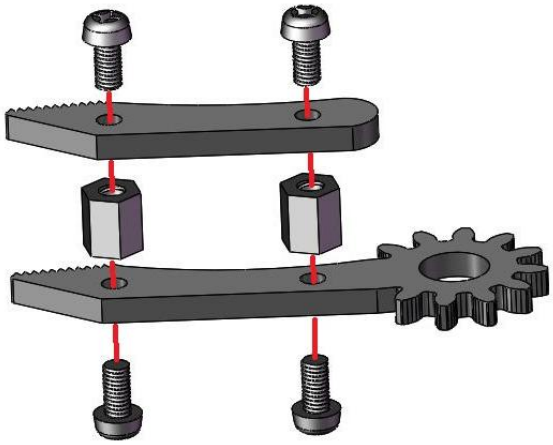
Acrylic Sheet 1PCS	Servo 1PCS	
		
Acrylic Sheet 2PCS	**Bag No.④** M2x10MM Screw 2PCS	**Bag No.⑧** M2 Nut 2PCS
		

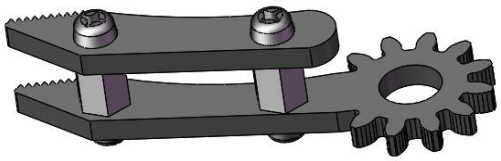




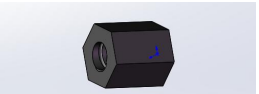
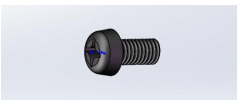
Step 11 Assemble the Left Clip Plate of The Clamp

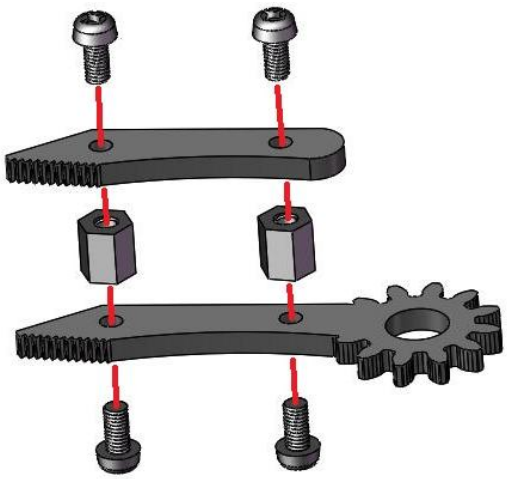
Acrylic Sheet 1PCS	Acrylic Sheet 1PCS	**Bag No.⑨** M3x7MM Standoff 2PCS	**Bag No.⑬** M3x6MM Screw 4PCS
			

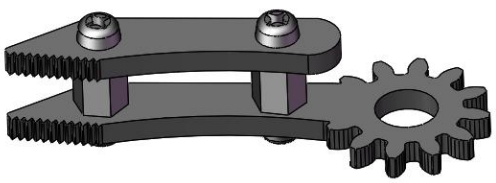




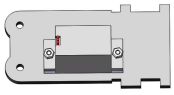
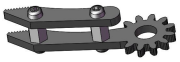
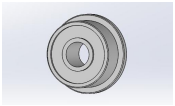
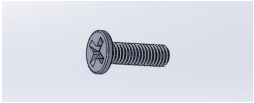
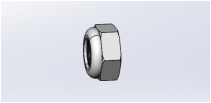
Step 12 Assemble the Right Clip Plate of The Clamp

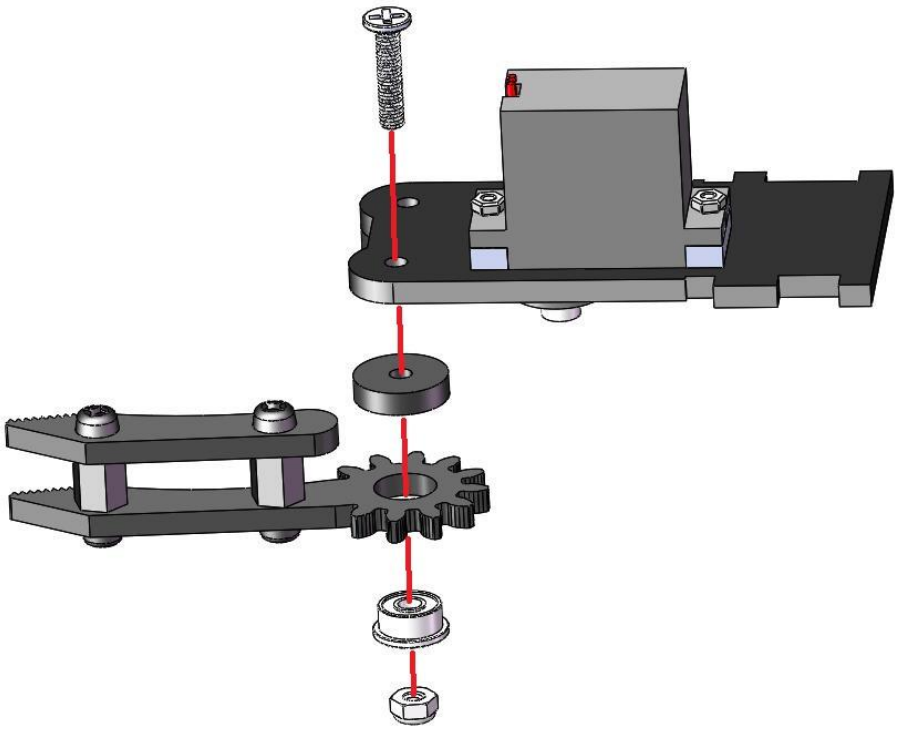
Acrylic Sheet 1PCS	Acrylic Sheet 1PCS	**Bag No.⑨** M3x7MM Nylon Column 2PCS	**Bag No.⑬** M3x6MM Nylon Screw 4PCS
			

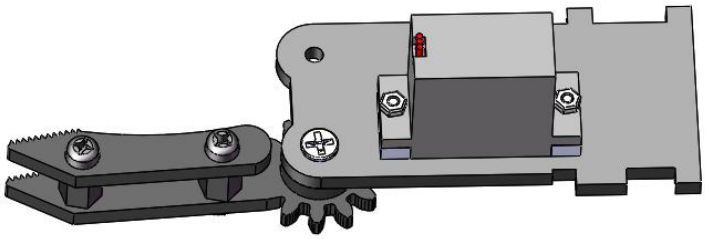




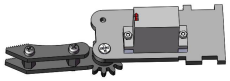
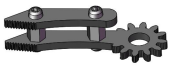
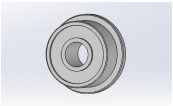
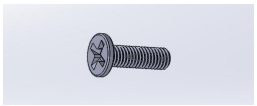
Step 13 Install the Left Clip Plate of The Clamp

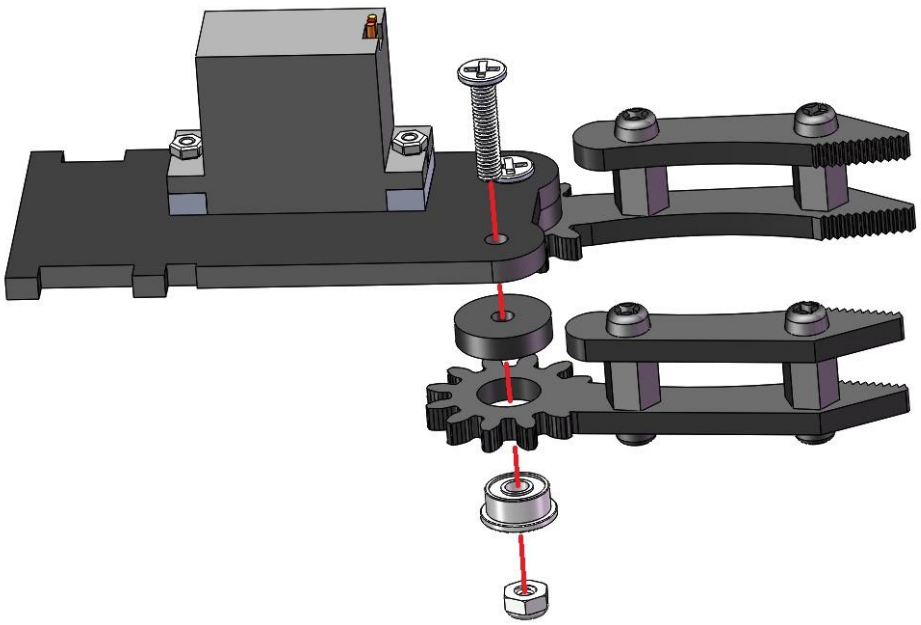
Step 10 Structure	Step 11 Structure	**Bag No.⑭** Flange Bearing 1PCS
		
Acrylic 1PCS	**Bag No.②** M3X14MM Screw 1PCS	**Bag No.⑦** M3 Lock Nut 1PCS
		

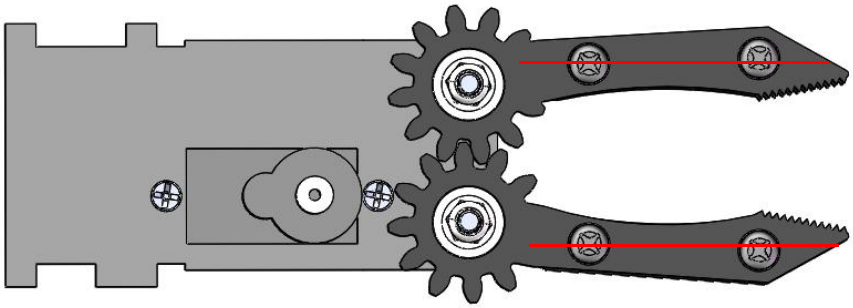




Step 14 Install the Right Clip Plate of the Clamp

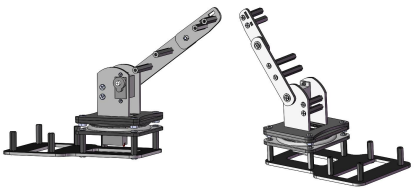
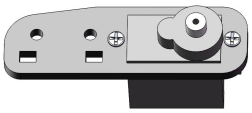
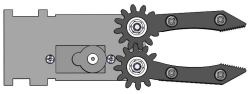
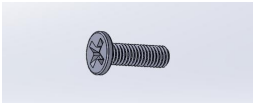
Step 13 Structure	Step 12 Structure	**Bag No.⑭** Flange Bearing 1PCS
		
Acrylic 1PCS	**Bag No.②** M3X14MM Screw 1PCS	**Bag No.⑦** M3 Lock Nut 1PCS
		

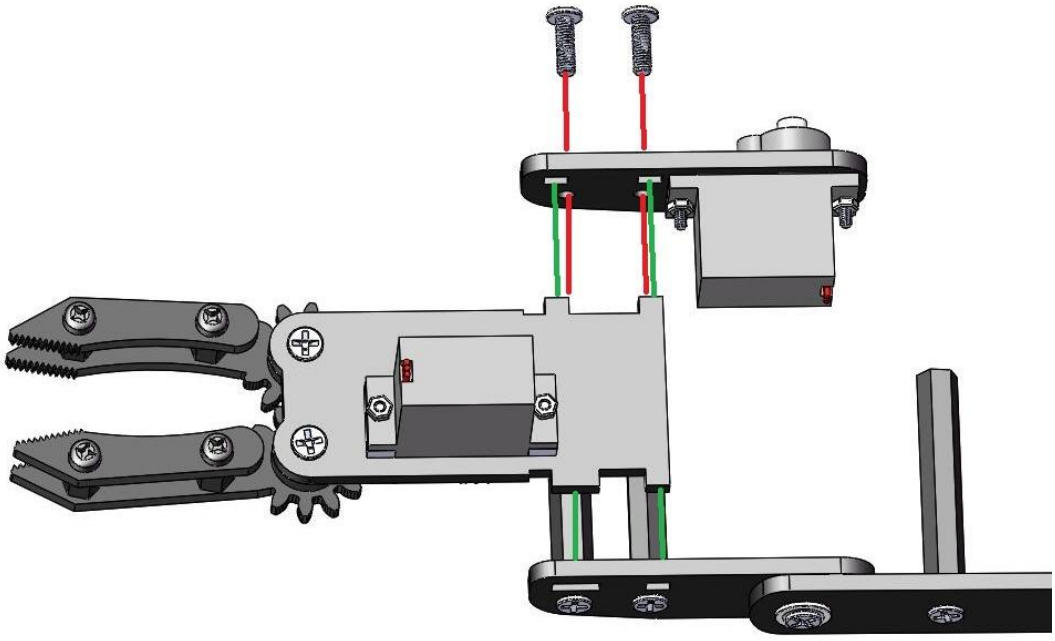


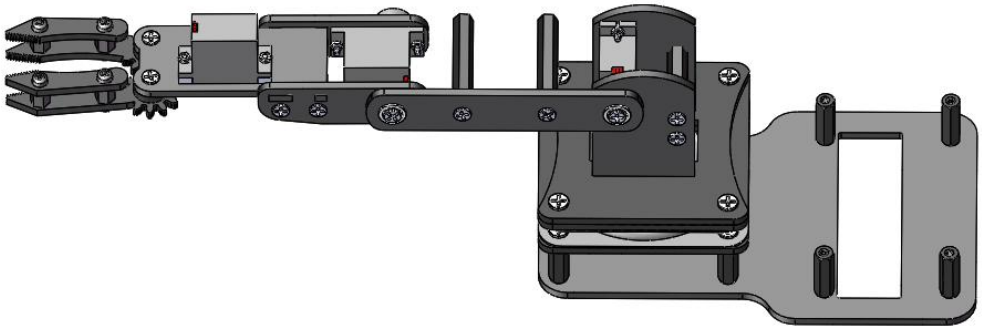


During installation, keep the screw threads horizontal

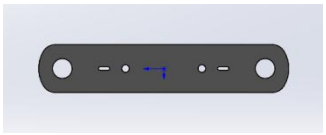
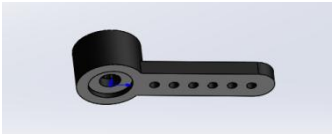
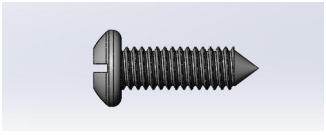
Step 15 Assemble the Clamp and Upper Arm

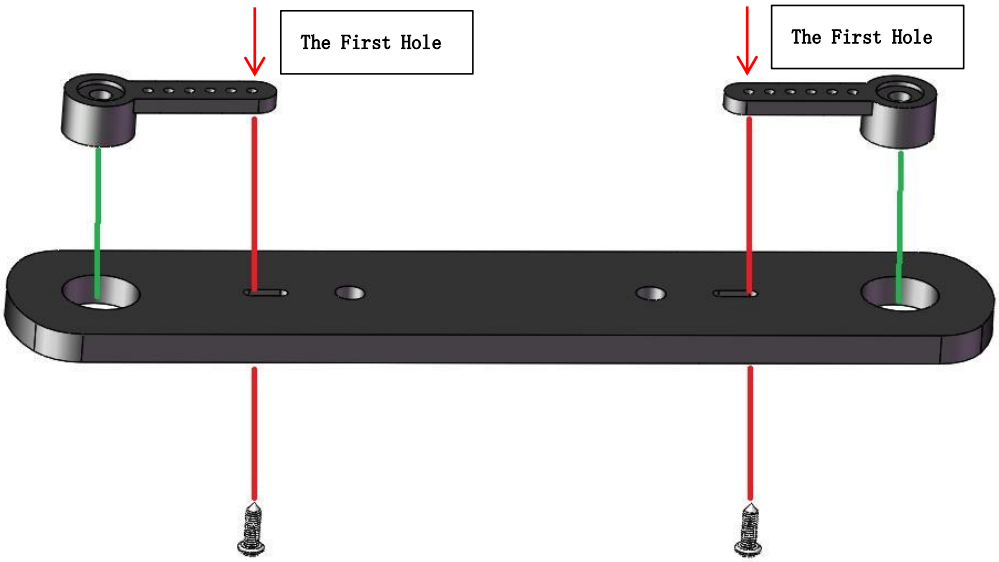
Step 8 Structure	Step 9 Structure
	
Step 14 Structure	**Bag No.③** M3X10MM Screw 2PCS
	



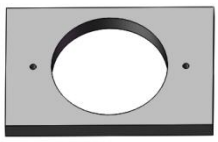
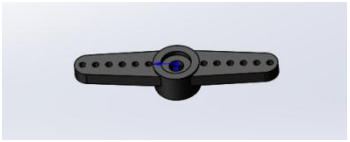
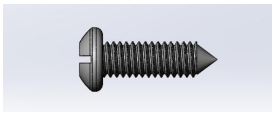


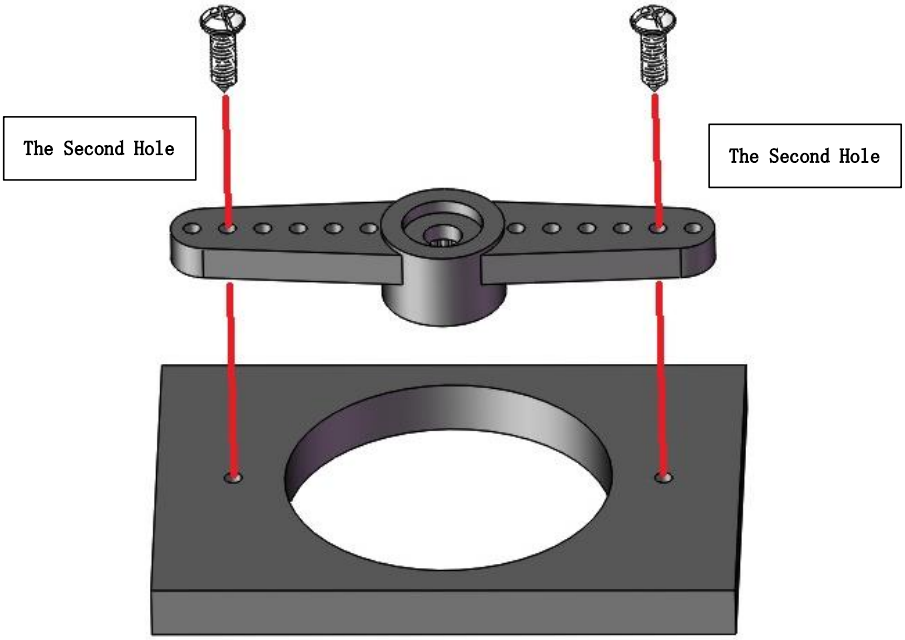
Step 16 Assemble the linkage between Servo B and Servo C

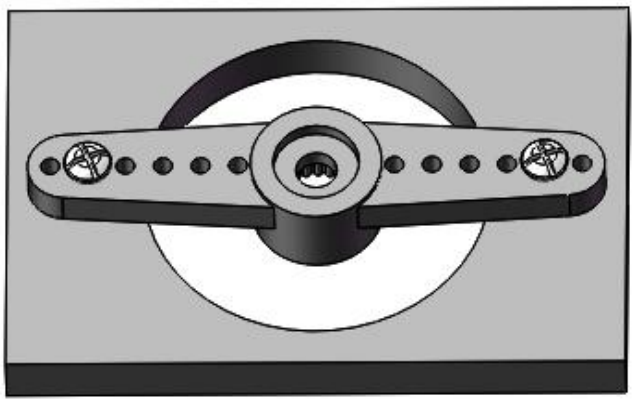
Acrylic Sheet 1PCS	Servo Arm 2PCS	**Bag No.⑤** M1.4X5 Lock Screw 2PCS
		



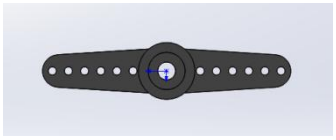
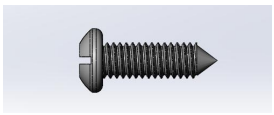
Step 17 Assemble the Mounting Structure for Servo A

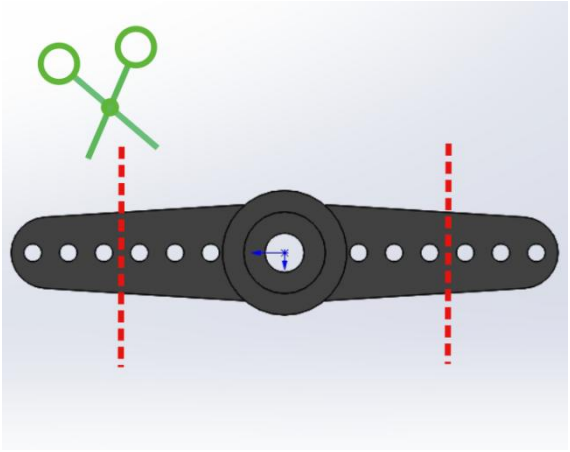
Acrylic Sheet 1PCS	Servo Arm 1PCS	**Bag No.⑤** M1.4X5 Lock Screw 2PCS
		



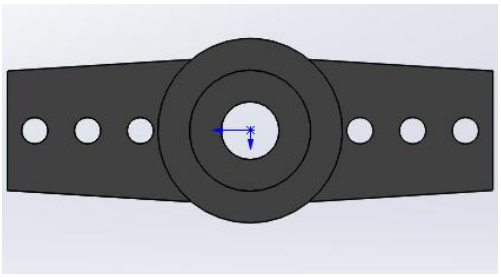


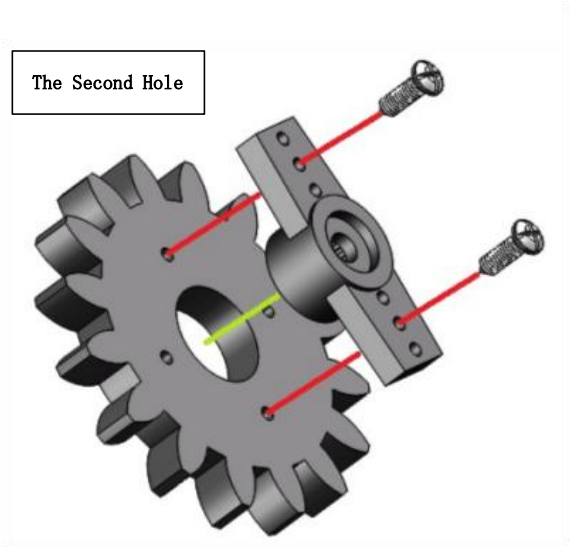
Step 18 Assemble the Mounting Structure for Servo D

Acrylic Sheet 1PCS	Servo Arm 1PCS	**Bag No.⑤** M1.4X5 Lock Screw 2PCS
		

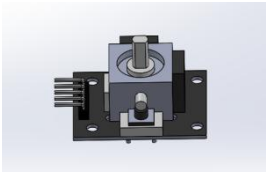
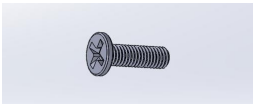


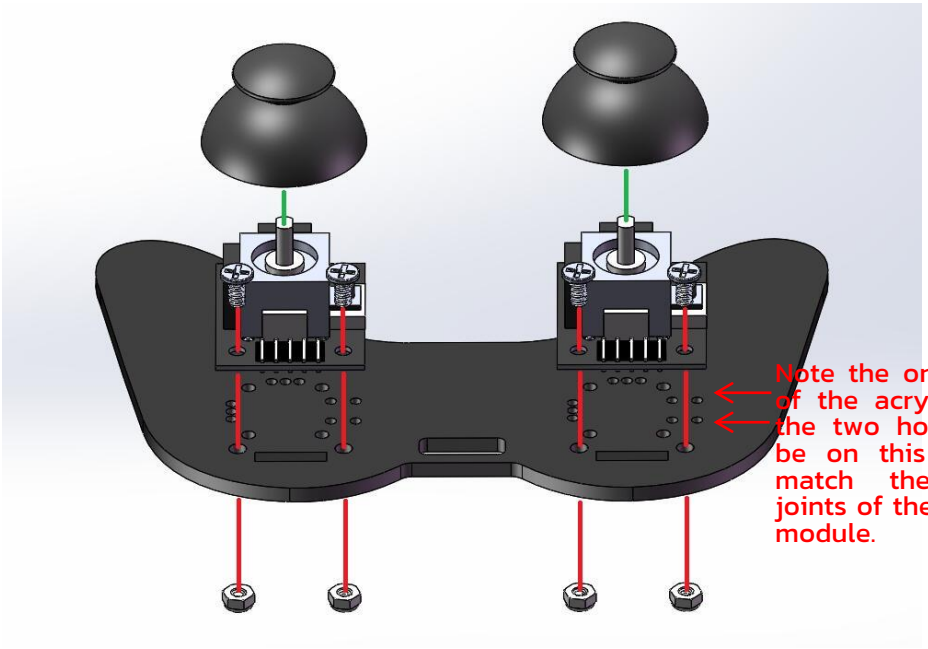
Pre-cut before factory shipment.

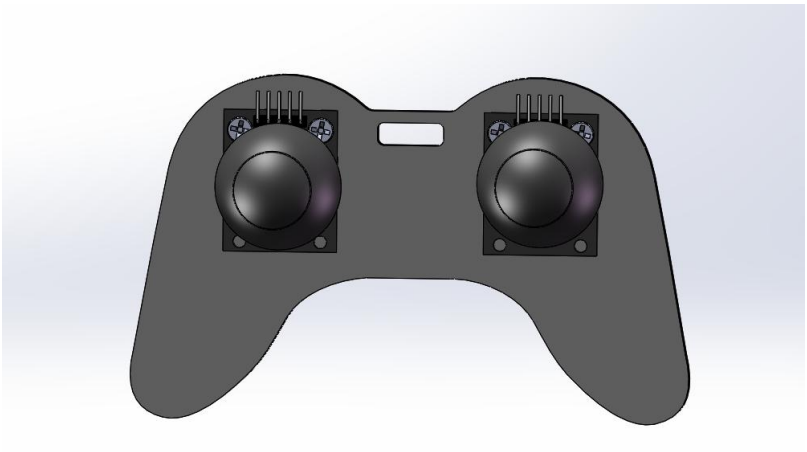




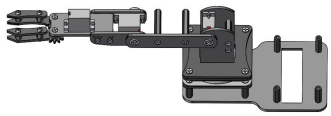
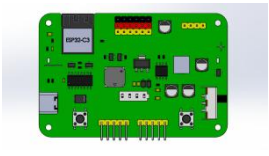
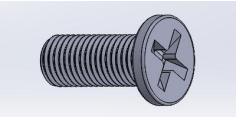
Step 19 Assemble the Joystick

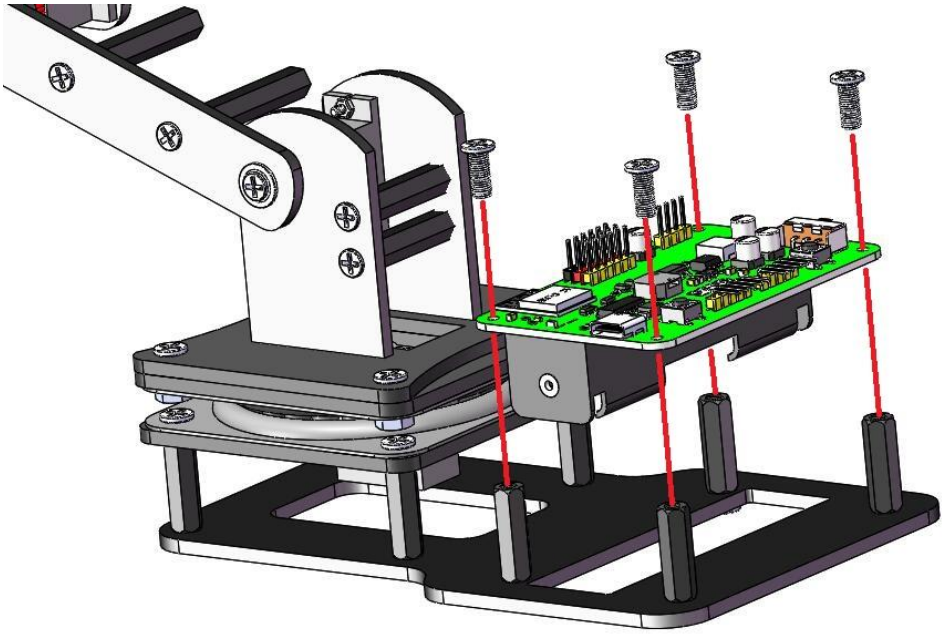
Acrylic Sheet 1PCS	Joystick Module 2PCS	
		
Joystick Hat 2PCS	**Bag No.③** M3X10MM Screw 4PCS	**Bag No.⑦** M3 Lock Nut 4PCS
		

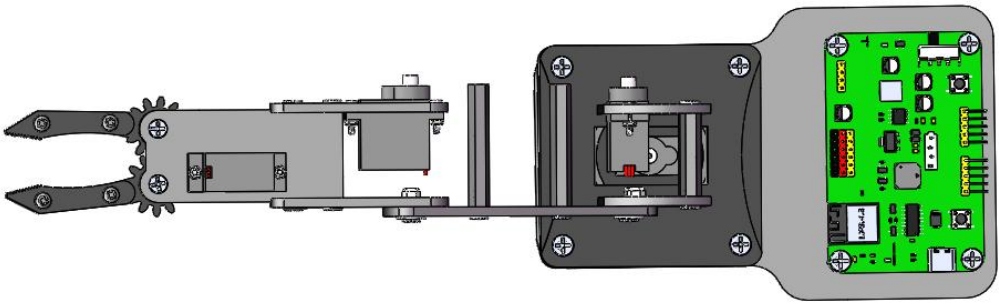




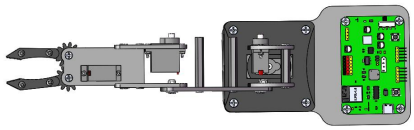
Step 20 Install the Esp32 Board

Step 15 Structure	ESP32 Control Board 1PCS	**Bag No.①** M4X10MM Screw 4PCS
		

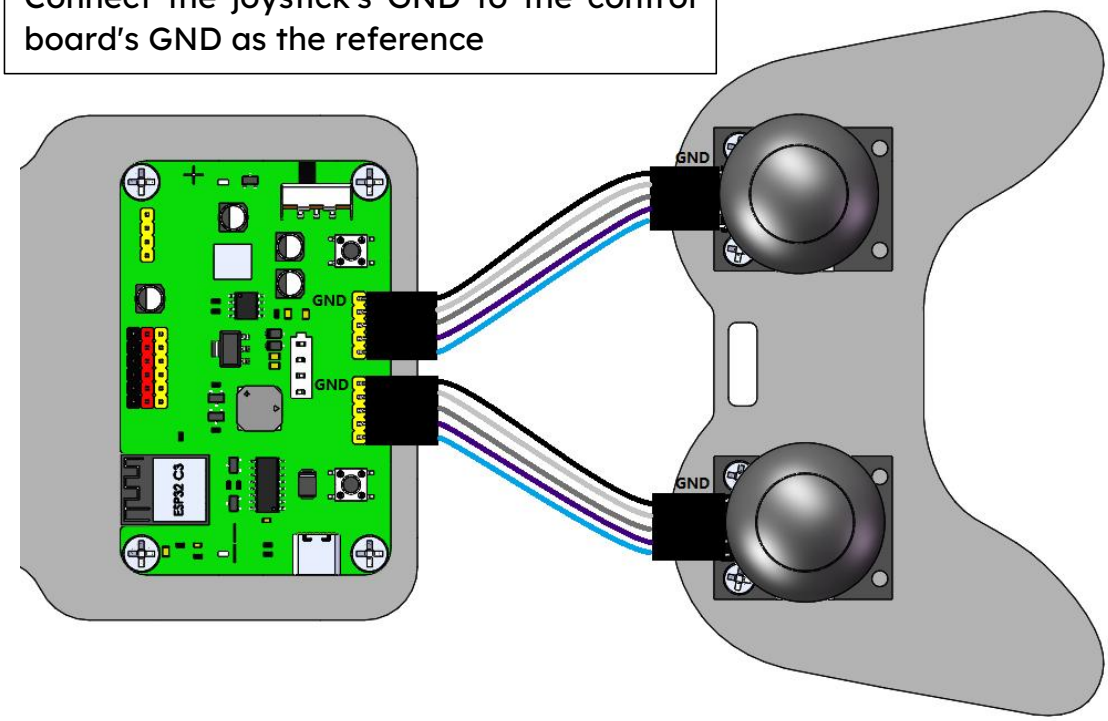


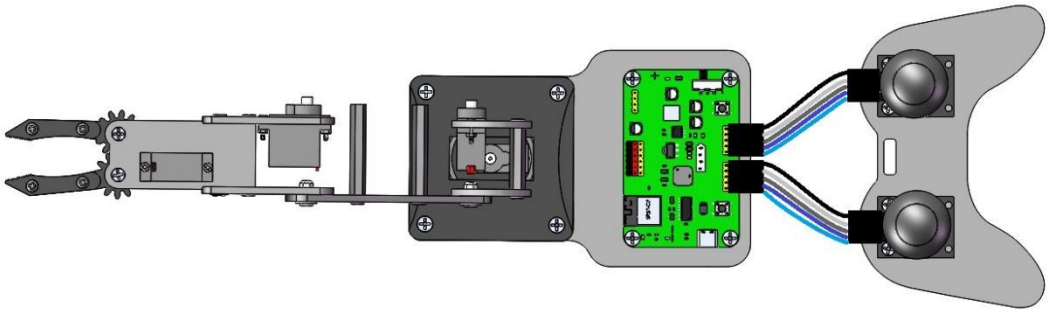


Step 21 Connect the Joystick

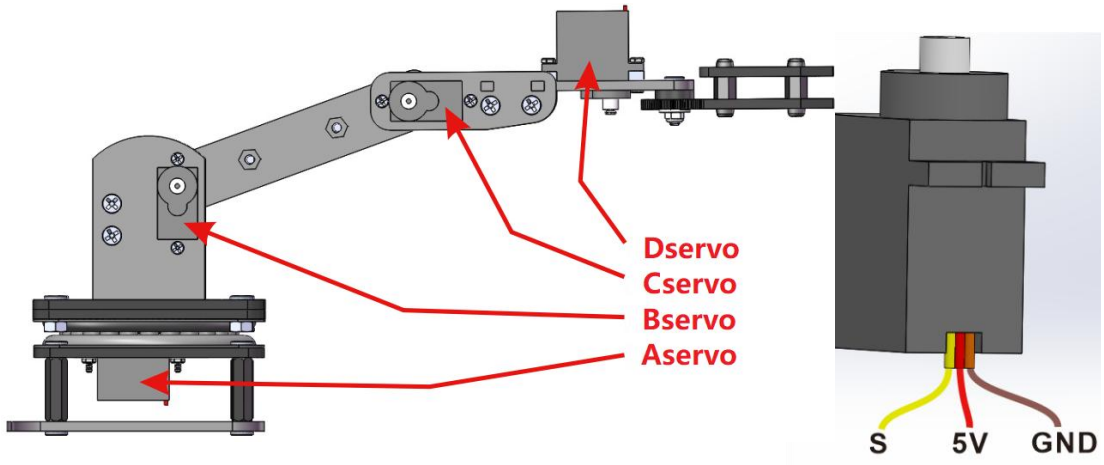
Part in Step 19	Part in Step 20
	
5P Dupont Wire 2PCS	
	

Connect the joystick's GND to the control board's GND as the reference

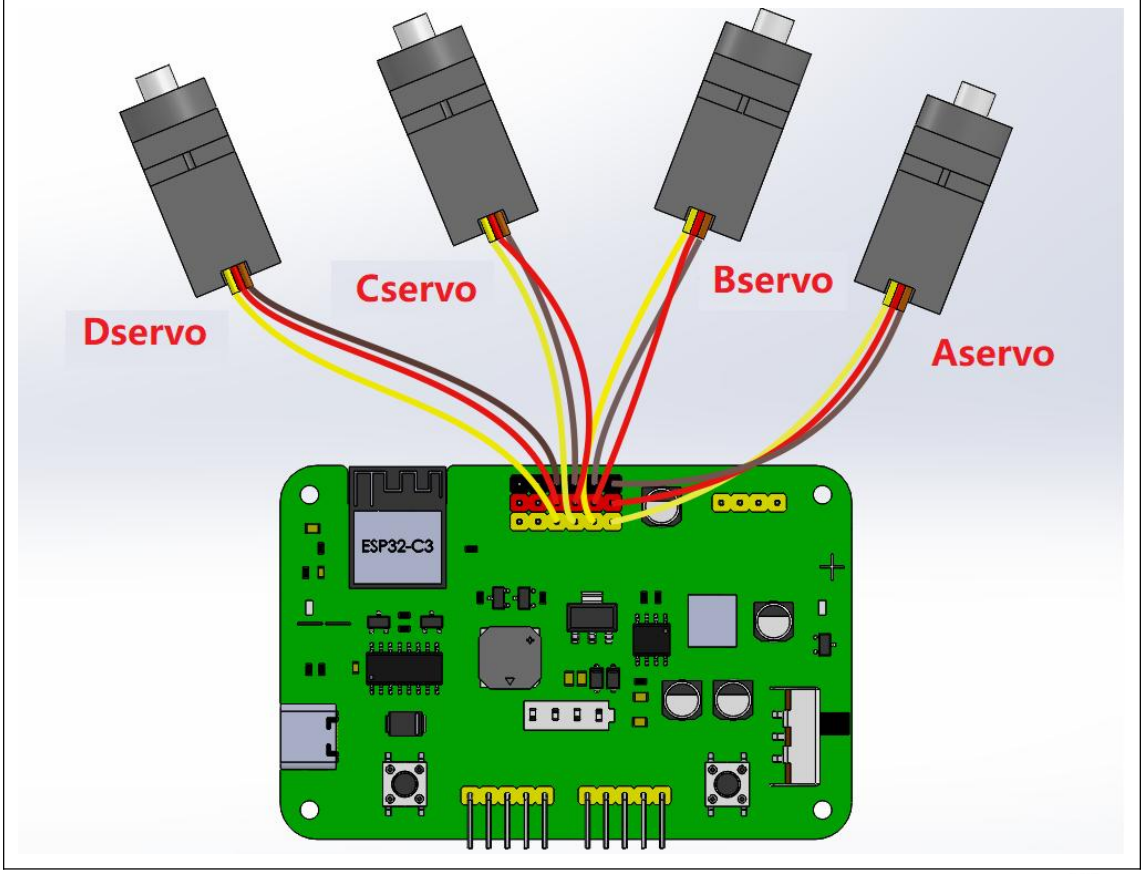


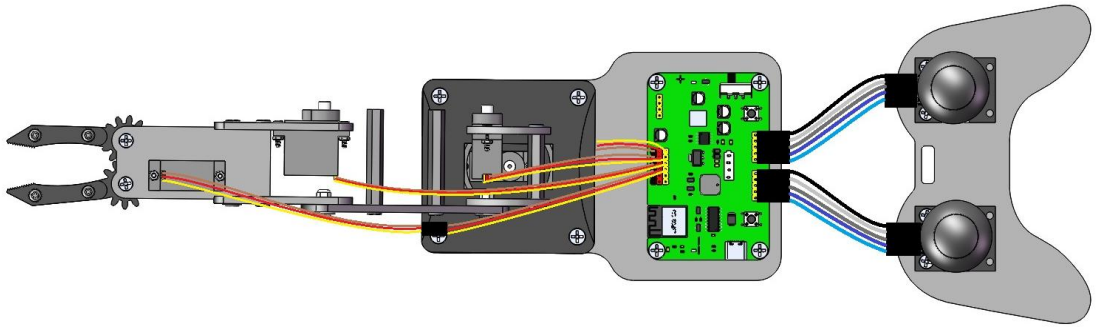


Step 22 Connect the Servos to the ESP32 Board



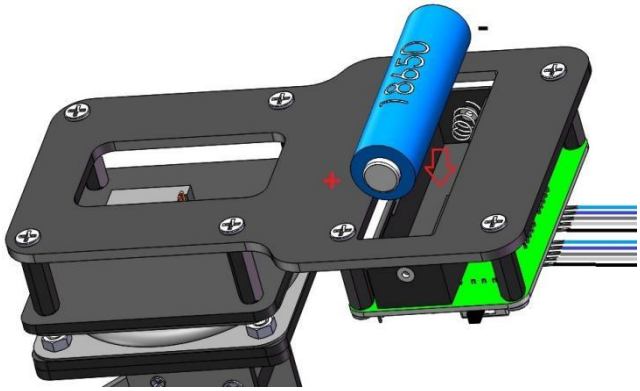
Dservo=GPIO7	Cservo=GPIO6	Bservo=GPIO5	Aservo=GPIO4



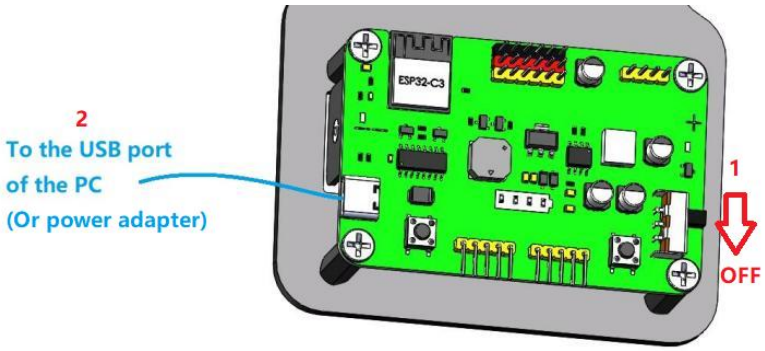


Step 23 Initialize the Servo Angle (important!)

Install a 18650 lithium battery (need to be purchased by yourself)

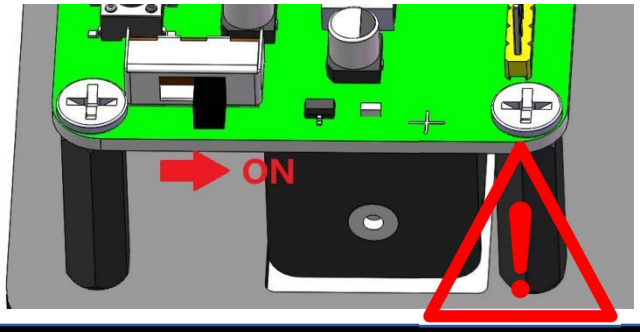


When installing the battery for the first time, please use the USB cable to charge the battery to activate the battery!



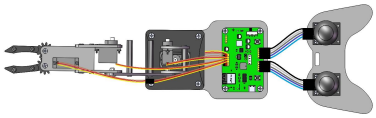
Very important: Turn on the power switch！ ！ ！

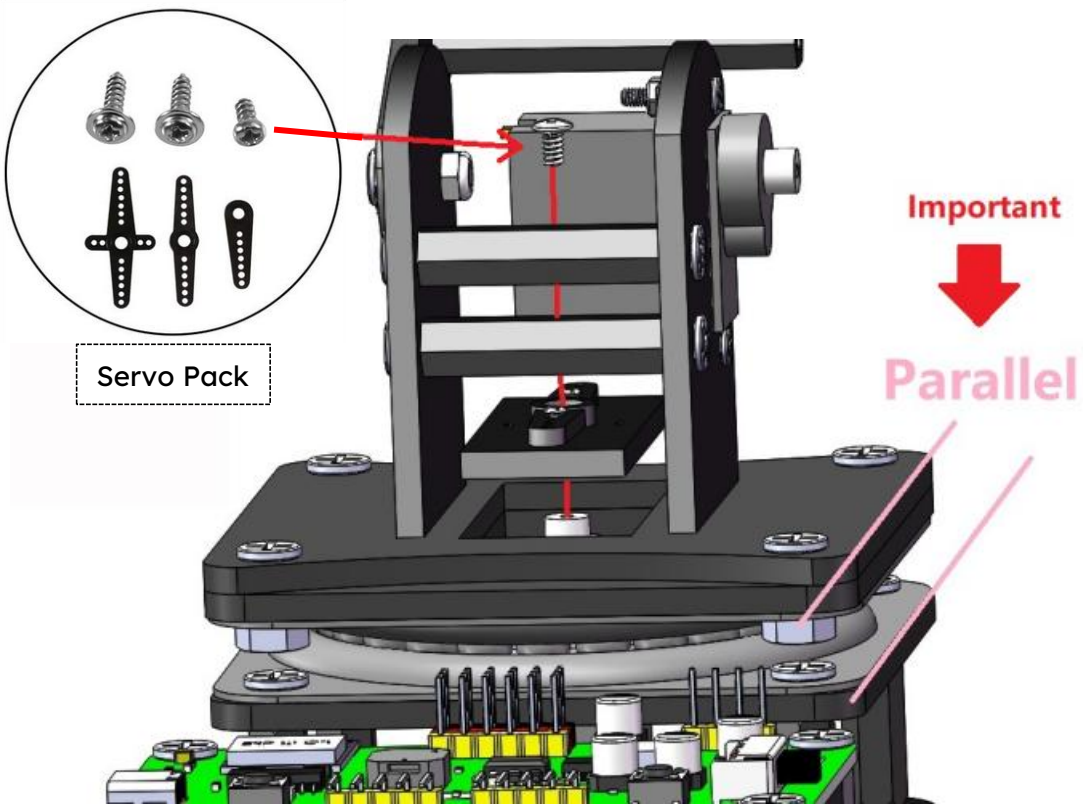
The firmware has been successfully programmed into the control board. When powered on, the board will automatically position all servo motors at the 90-degree default position.

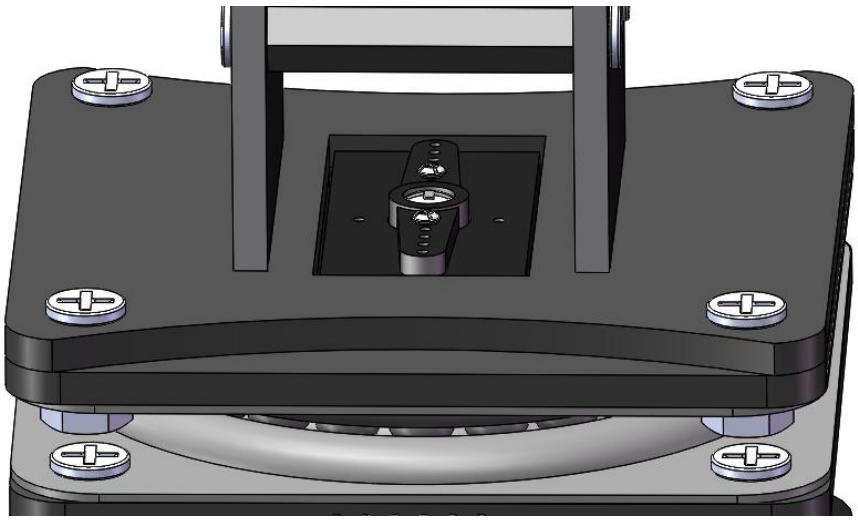


1. Make sure the battery is fully charged!
2. When you turn on the power switch, You will hear the servo motor shaft rotating.
3. In the following installation steps, please keep the power on to prevent the servo angle from being changed!

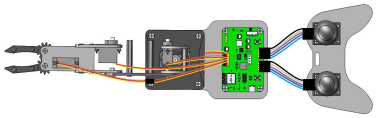
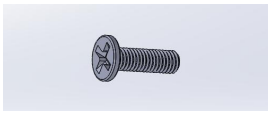
Step 24 Fix the Robot Arm to Servo A of the Base

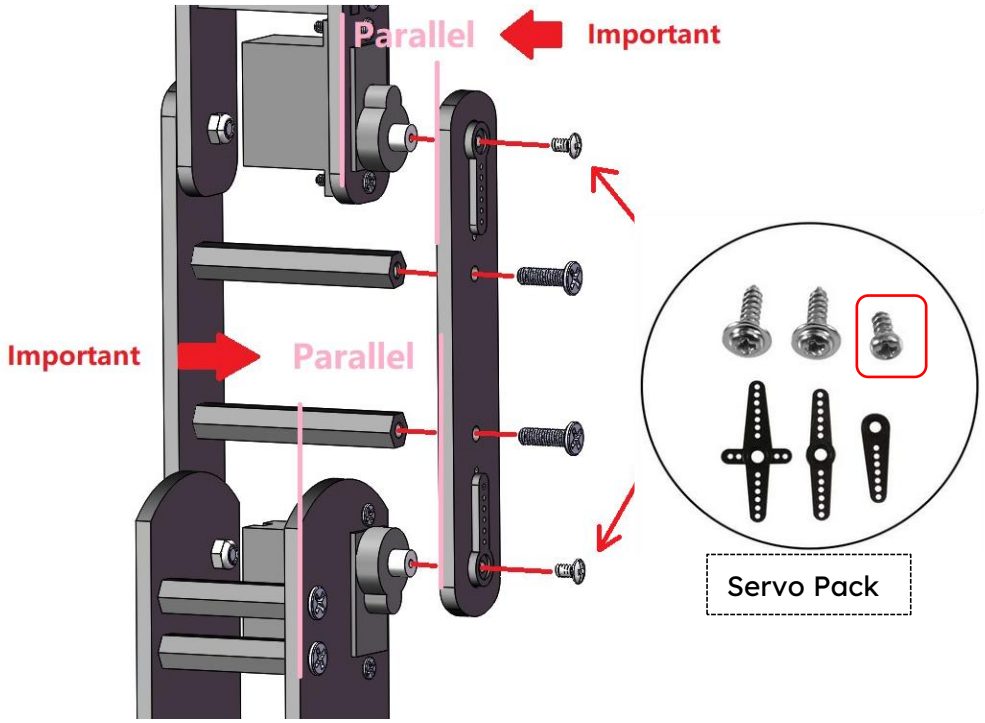
Step 23 Structure	Step 17 Structure	M2.5X4mm Screw 1PCS
		

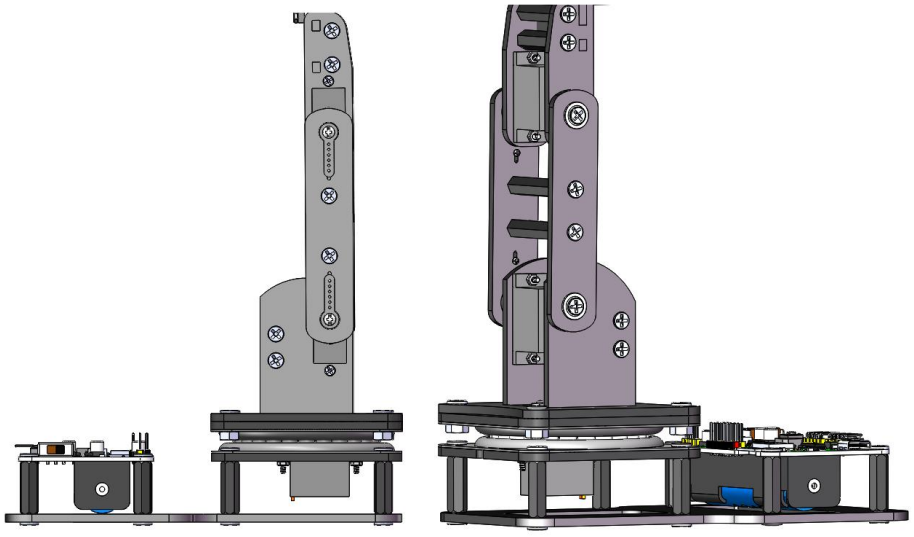




Step 25 Assemble the Middle Arm

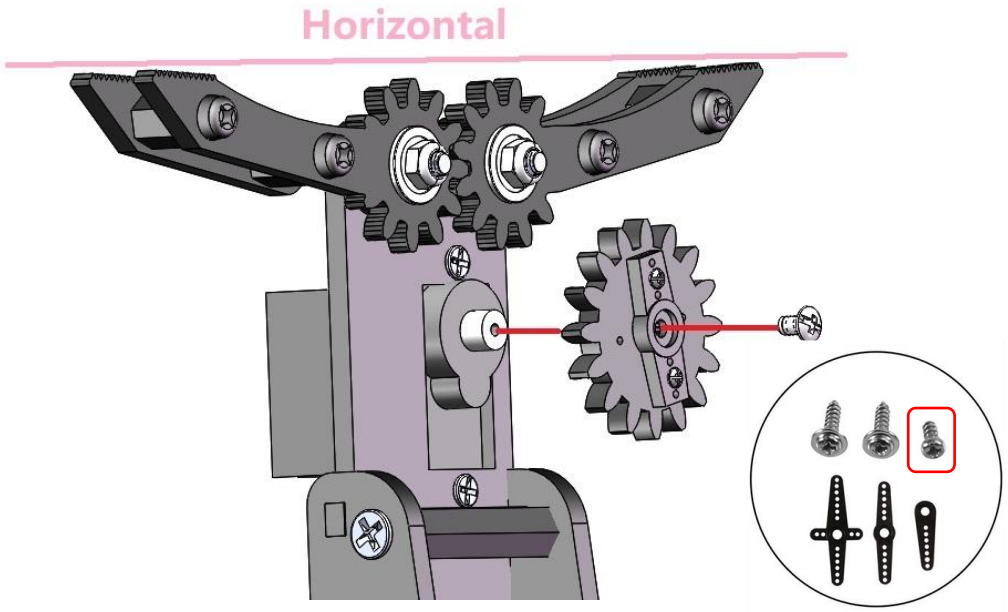
Step 24 Structure	Step 16 Structure
	
Bag No.③ M3X10MM Screw 2PCS	M2.5X4mm Screw 2PCS
	

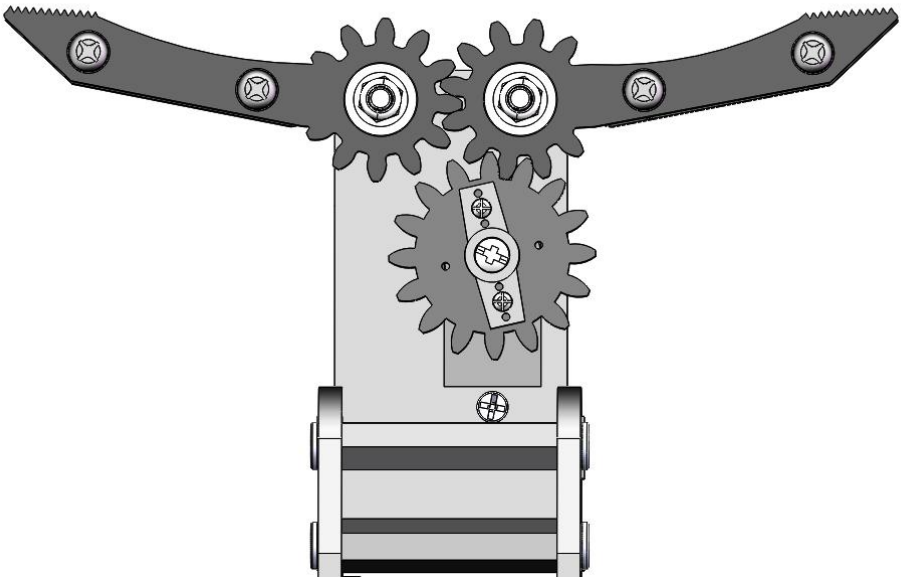




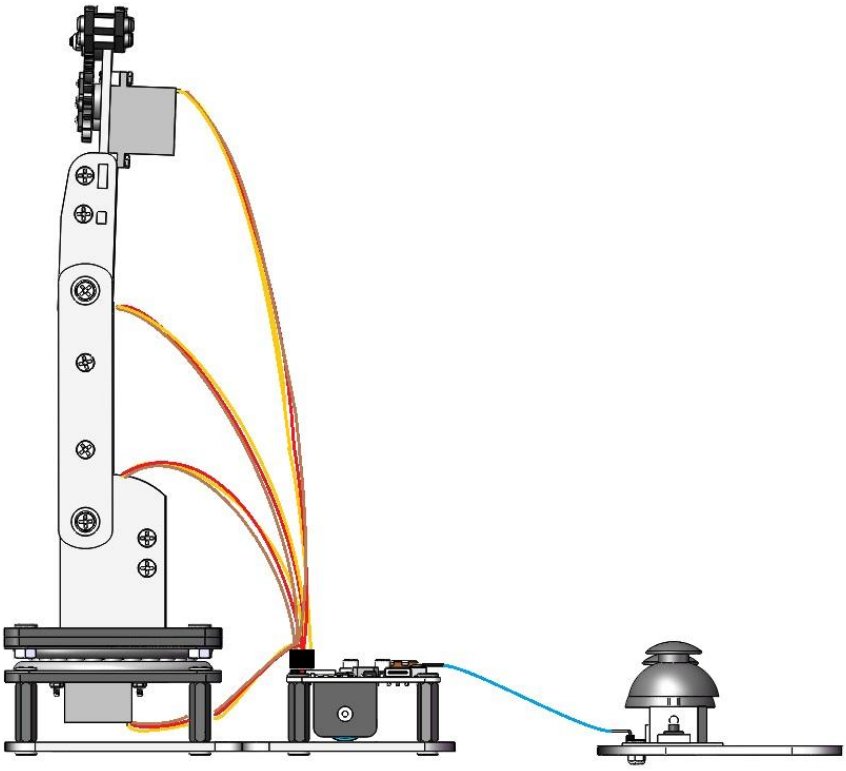
Step 26 Fix the Clip Plates to Servo D

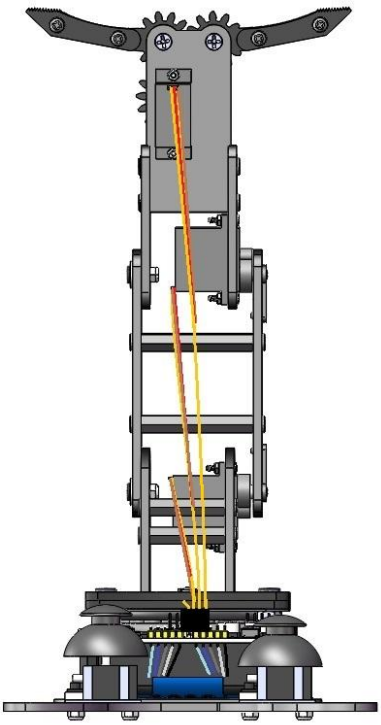
Step 25 Structure	Step 18 Structure	M2.5X4mm Screw 1PCS
		





Step 27 Assembly completed





3

Using the Robot Arm

3.1 Robotic Arm Safety and Operating Guidelines

To ensure optimal performance and maximize the service life of your ESP32 Robotic Arm, please read and follow these guidelines carefully.

1. Primary Operating Precautions

Do Not Overload: This arm is designed for lightweight tasks. The maximum recommended load for the gripper is 40 grams.

Maintain a Clear Workspace: Ensure the area around the robotic arm is free of obstacles to allow unobstructed movement of all joints.

Avoid Forced Movement: Never manually force or twist any joint while the arm is powered on, as this may damage internal gears and servo motors.

2. Critical Gripper Maintenance (Servo Protection)

Continuously gripping an object under load places significant stress on the servo motor (particularly the D-axis driving the gripper), causing it to generate heat. Therefore, we strongly recommend limiting continuous load-bearing gripping to no more than 40 seconds. After completing a gripping operation, release the object immediately by opening the gripper to allow the motor to cool down and rest. Exceeding this duration is the primary cause of motor overheating and burnout.

Auto-Protection Function: Our latest optimized firmware includes a protective feature. If the gripper servo does not receive a new command within 40 seconds, it will automatically disengage (cease PWM signal transmission) to prevent overheating.

Best Practice: Actively control the gripper. When not manipulating objects, open the gripper or send commands to maintain it in a relaxed state.

3. Routine Maintenance

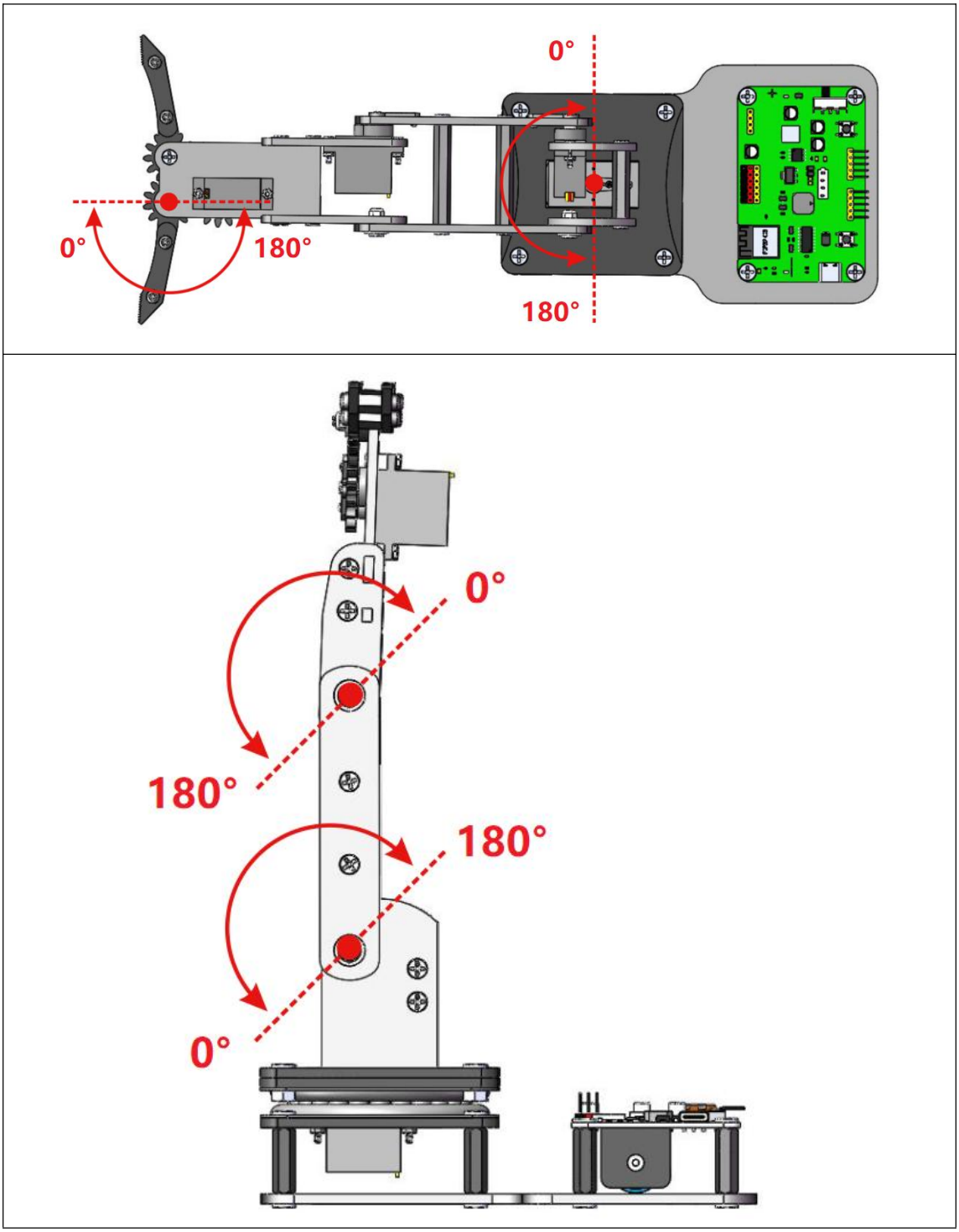
Power off the unit when not in use.

Periodically check and tighten all screws and connections.

Operate the robotic arm on a stable, flat surface.

Following these instructions will help ensure a reliable and durable robotic experience.

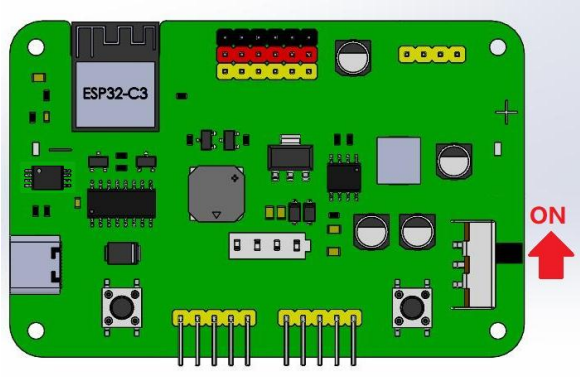
3.1 Degrees of Freedom

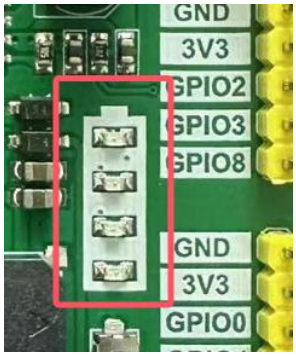


3.2 Control the eArm with a Joystick

Turn the eArm's power switch to the **ON** position.

■ 1: Make sure the 18650 lithium battery is installed.
■ 2: The battery indicator shows 3 bars or more.

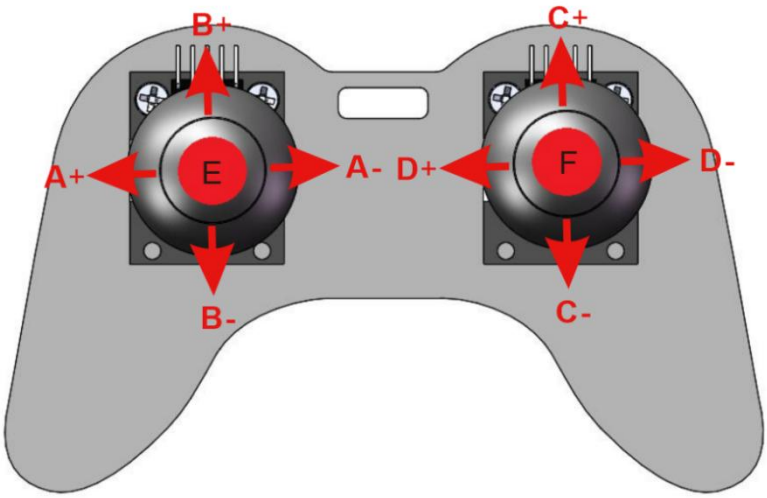




After powering on the robot, the default control mode is **Joystick Mode**. To switch between control modes, press the **"F"** button on the right joystick. Available modes include:

- ✓■ Joystick Mode (default)
- ✓■ Web App Mode

When you press "**F**", the buzzer will beep once, indicating the switch to **Web App Mode**. In this mode, joystick controls will be **disabled**.



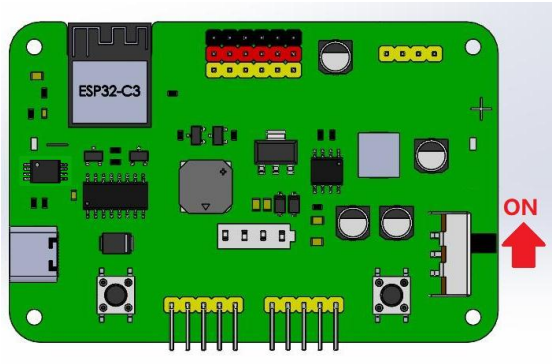
A+: Rotate arm left (Servo A)	A-: Rotate arm right (Servo A)
B-: Raise rear arm (Servo B)	B+: Lower rear arm (Servo B)
C-: Raise the forearm (Servo C)	C+: Lower the forearm (Servo C)
D+: Open the gripper (Servo D)	D-: Close the gripper (Servo D)

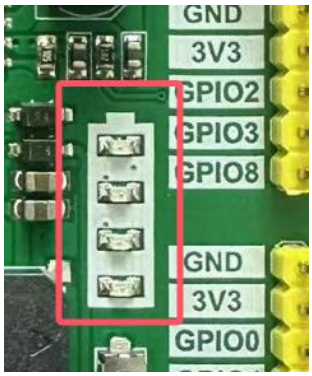
F: Toggle between Joystick and Web App control modes
E: Enable/disable the buzzer

3.3 Control the eArm with Web App

Turn the eArm's power switch to the ON position.

■ 1: Make sure the 18650 lithium battery is installed.
■ 2: The battery indicator shows 3 bars or more.

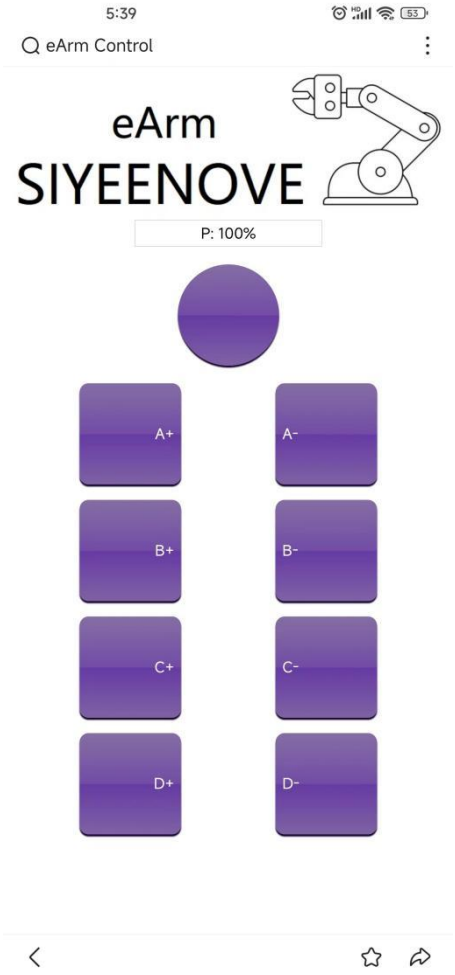




Turn on the phone's Wi-Fi and connect to the network named 'eArm'
■ : Once connected to the Wi-Fi, you may see a 'Network connection failed' message; just disregard it.
■ : Step 1: Turn off mobile data in your phone settings
■ : Step 2: This ensures stable connection to the Web App

1. Open the web browser on your phone
2. Type 192.168.4.1 into the address bar
3. Press Enter to connect

After successful connection, the control interface will appear in your browser - you can now operate the eArm!



Button	Mechanical Movement
A+	Robot arm turns left
A-	Robot arm turns right
B-	Rear arm raises
B+	Rear arm lowers
C-	Front arm raises
C+	Front arm lowers
D+	Gripper opens
D-	Gripper closes

4

Programming Learning

4.1 Before You Begin: Important Notes

The ESP32-C3 control board comes preloaded with firmware that enables robotic arm operation as demonstrated in the previous section. This functionality will remain available until you upload new code.

Important notes about code uploading:

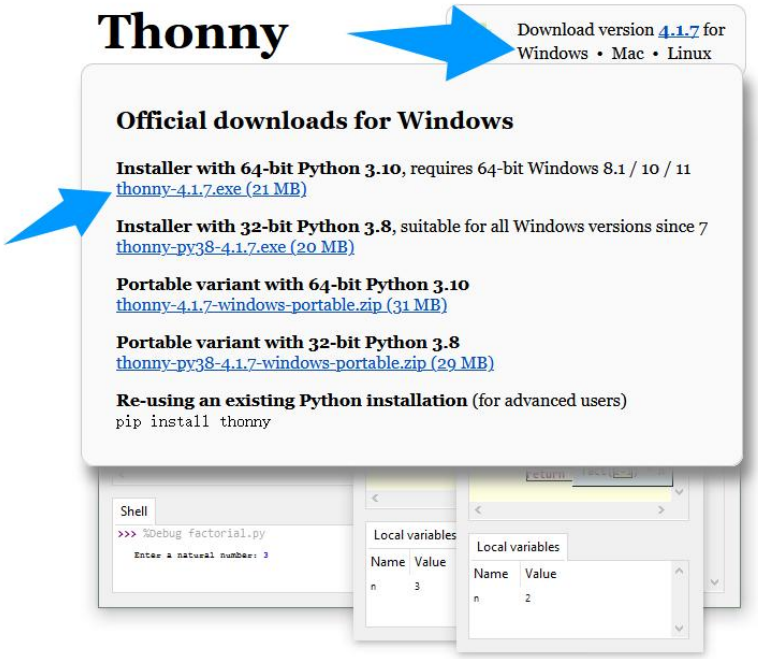
- 1: The ESP32-C3 board always executes the most newly uploaded program
- 2: Each new code upload overwrites the previous program

This section will guide you through programming the robot using code examples ranging from basic to advanced. The final example contains our factory-preloaded firmware - the same code that enables the arm functionality shown in the previous section.

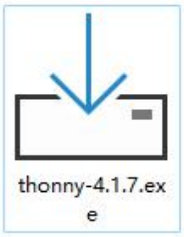
4.2 Install Thonny IDE

The latest version of Arduino IDE can be downloaded from the Arduino official website：https://thonny.org/

In the following we will demonstrate the installation of **Windows Win 11 and newer, 64 bits** version IDE on PC.



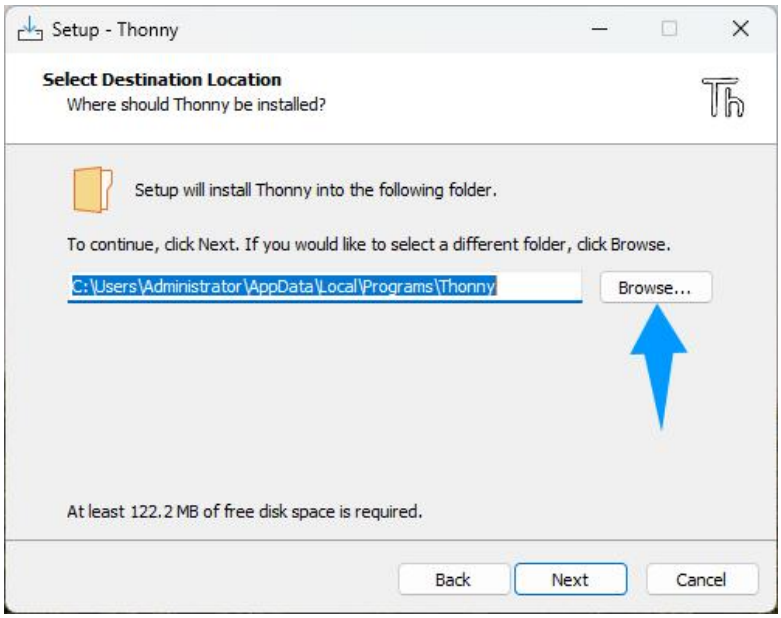
After the installation package is downloaded, double-click the file to install the software.



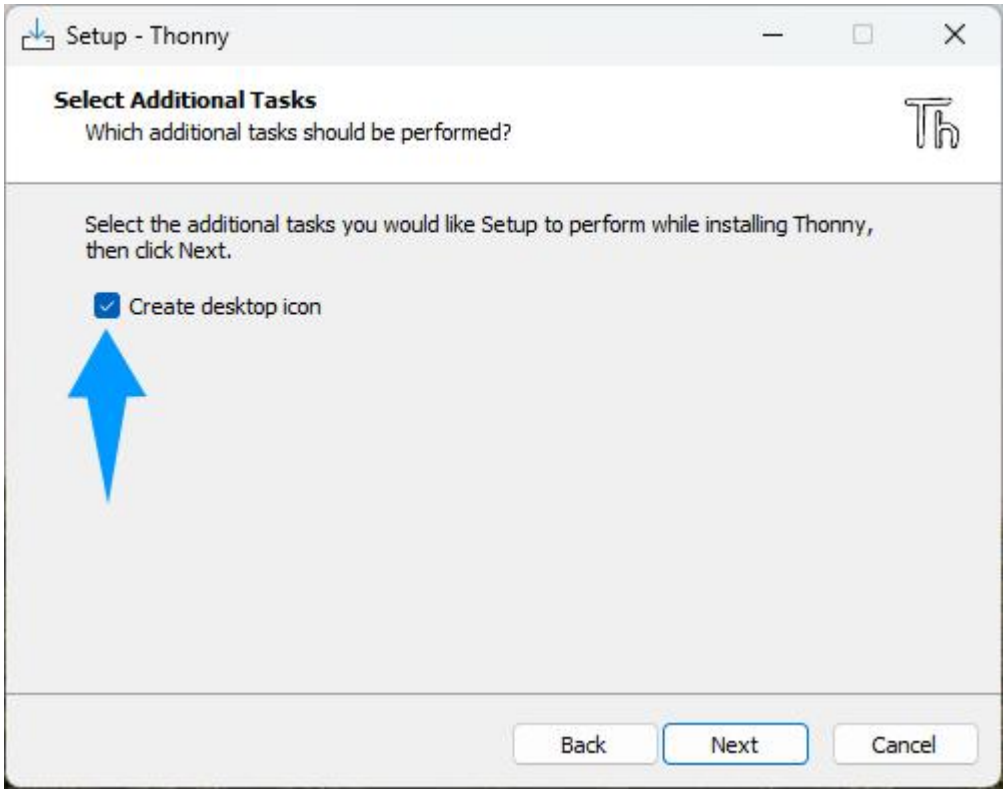
If you're not familiar with computer software installation, you can simply keep clicking "Next" until the installation completes.

If you want to change Thonny's installation path, you can click "Browse" to modify it. After selecting installation path, click "OK".

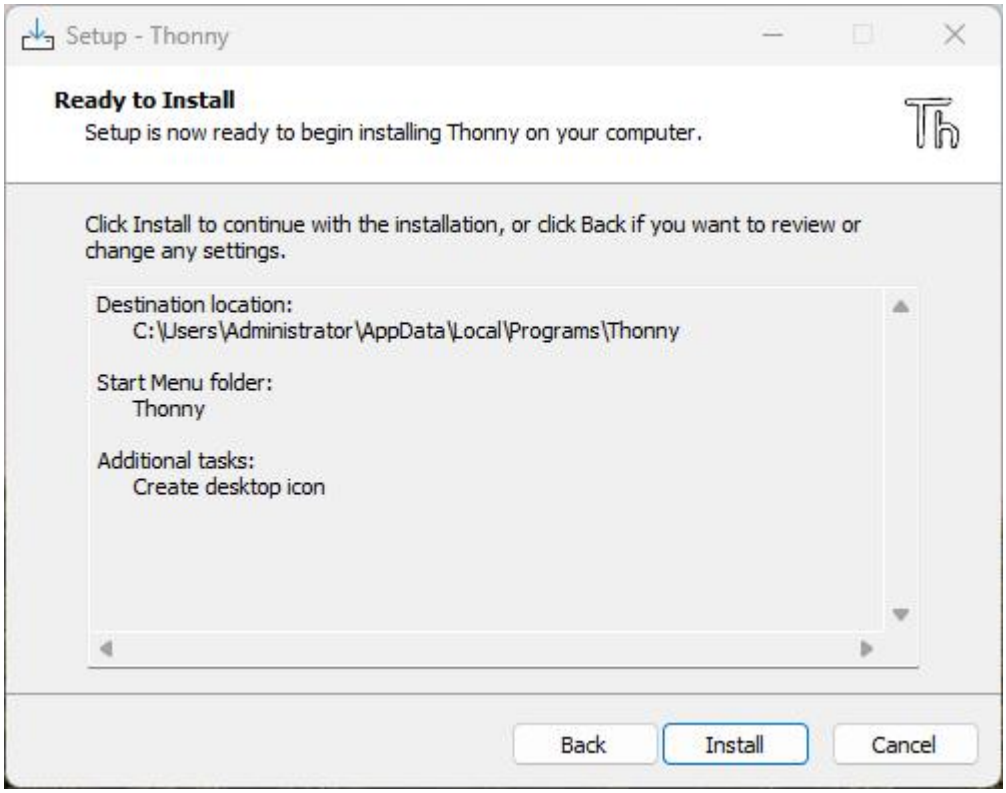
If you do not want to change it, just click "Next".



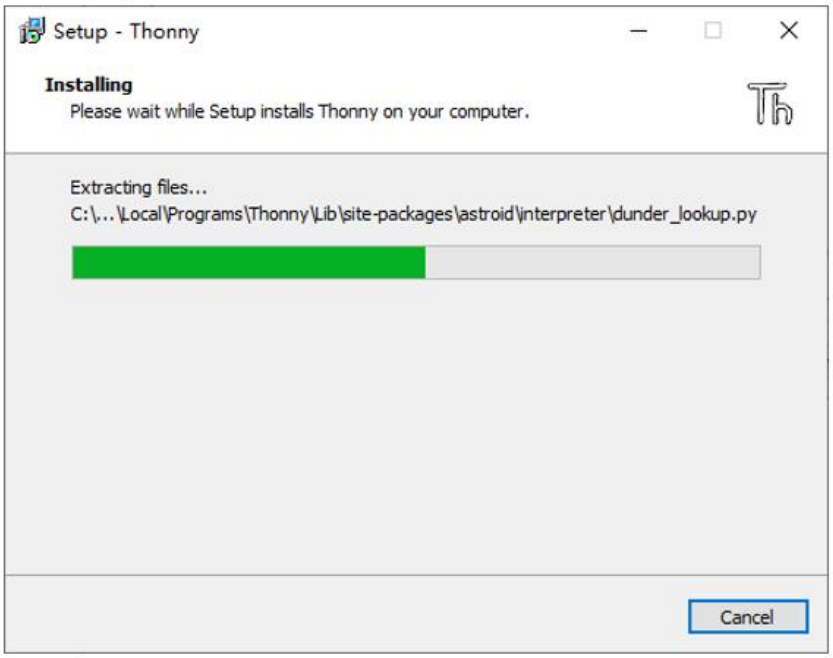
Check “Create desktop icon” and then it will generate a shortcut on your desktop to facilitate you to open Thonny later.



Click “install” to install the software.



During the installation process, you only need to wait for the installation to complete, and you msut not click “Cancel”, otherwise Thonny will fail to be installed.

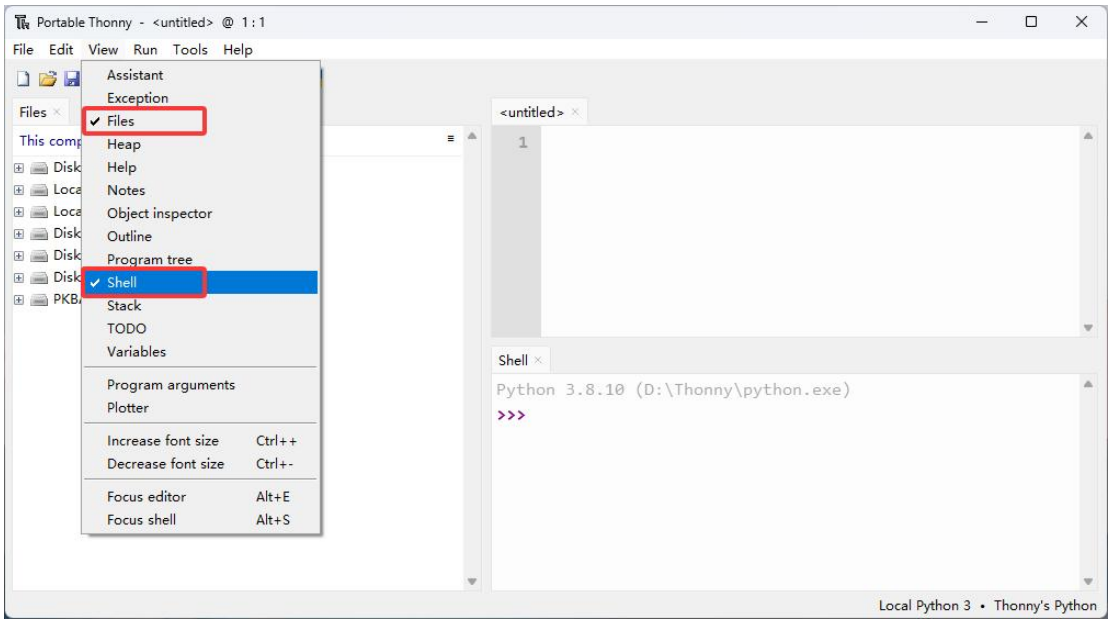


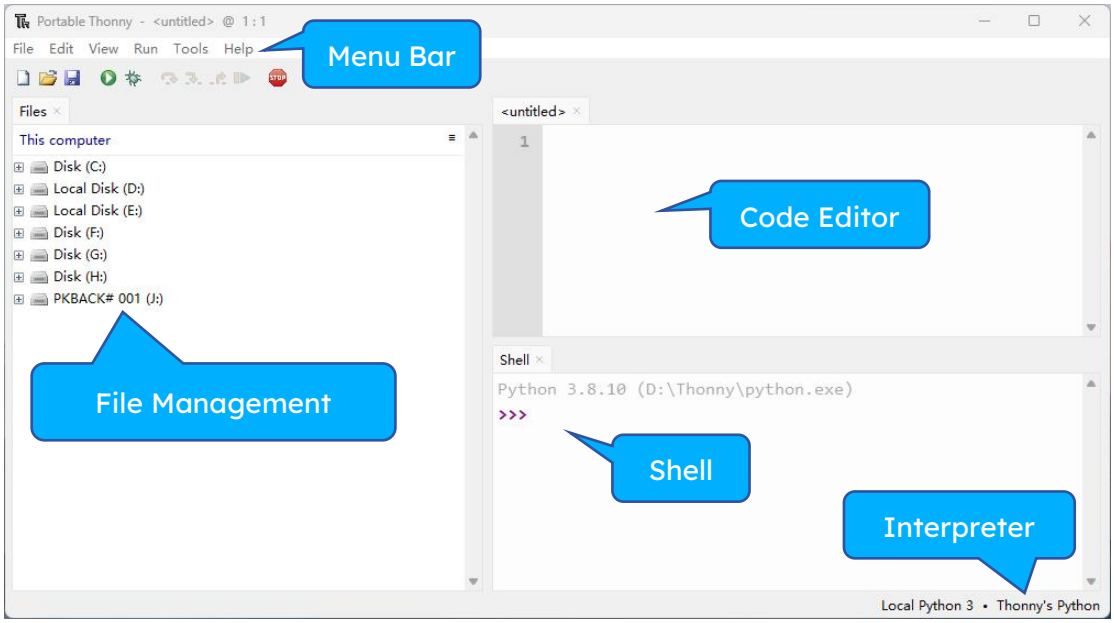
Once you see the interface as below, Thonny has been installed successfully.

If you’ve check “Create desktop icon” during the installation process, you can see the below icon on your desktop.

4.2 Basic Configuration of Thonny

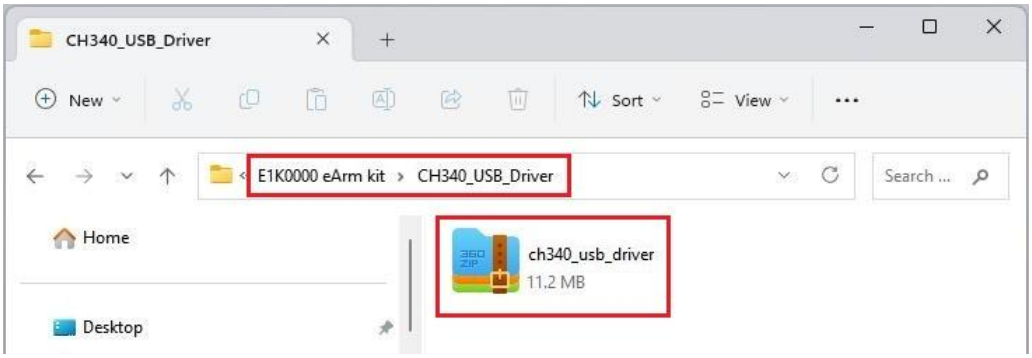
Click the desktop icon of Thonny and select “View” -> “Files” and “Shell”.





4.3 Install CH340 USB driver

Driver file is provided in our tutorial package:



4.3.1 Install the Driver on Windows system

Unzip the file we provided and double-click **"SETUP"** to run the installer:

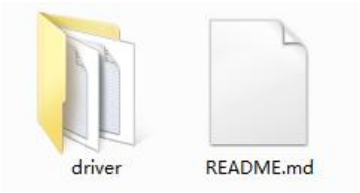
If the driver is successfully installed, when the esp32 board is connected to the computer through the USB cable, under the path **"Computer" -> "Properties" -> "Device manager"**, you can find **"USB-SERIAL CH340 (COM3)"**, as shown below:

Note: Driver display symbol is "USB-SERIAL CH340 (COMx)", "x" can be any number, it depends on what number your computer assigns to the ESP32 device.

4.3.2 Install the Driver on Linux system

Drivers are almost certainly built into your Linux kernel already and it will probably just work as soon as you plug it in. If not you can download the Linux CH340 Driver (but I'd recommend just upgrading your Linux install so that you get the "built in" one).

If you must install it by yourself, check the "README.md" file in the zip package.



4.3.3 Install the Driver on MAC system

Link to download the driver:

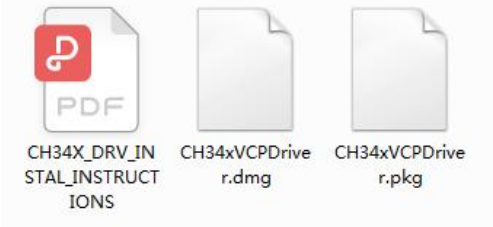
http://www.wch-ic.com/downloads/CH341SER_EXE.html

download

relation files

file name	file content
CH341SER.ZIP	CH340/CH341 USB to serial port Windows driver, supports Windows XP/Vista/7/8/8.1/10/11/ SERVER 2003/2008/2012/2016/2019/2022 -32/64bit, Microsoft WHQL Certified, supports USB to 3-line and 9-line serial port.
CH341SER_LINUX.ZIP	Linux driver for USB to serial port, supports CH340 and CH341, supports 32/64-bit operating systems.
CH341SER_MAC.ZIP	For CH340/CH341/CH342/CH343/CH344/CH347/CH9101/CH9102/CH9103/CH9104/CH9143, USB to serial port VCP vendor driver of macOS , supports OS X 10.9~10.15 , OS X 11 (Big Sur) and above , contains installation guide documents.
CH340DS1.PDF	CH340 datasheet, USB bus converter chip which converts USB to serial port, printer port, IrDA SIR etc. Integrated clock, supports various operating systems, chip information can be customized, this datasheet is about USB to serial port and USB to Infrared Adapter SIR.
CH341DS1.PDF	CH341 datasheet, USB bus converter chip with multiple communication interfaces, such as USB to serial port/parallel port/printer port/I2C interface etc. Drivers support for Windows/Linux/Android/Mac, etc. The datasheet is the description of USB to serial port and printer port.
CH341SER_ANDROID.ZIP	CH340/CH341/CH342/CH343/CH344/CH347/CH9101/CH9102/CH9103/CH9104/CH9143 USB to serial port Android driver-free application library and software, for Android OS 4.4 and above in USB Host mode, without loading Android kernel driver and without root access operation. Includes apk installer, lib library file (Java Driver), App Demo routines (USB to UART Demo project SDK).

Download the driver from the website and unzip the file to a local installation directory. You will get a PDF installation guide and two installation packages in different formats. For details, see the PDF installation guide.



4.4 Burning Micropython Firmware (Important)

To run Python programs on ESP32-C3, we need to burn a firmware to ESP32-C3 first.

4.4.1 Download MicroPython Firmware

Official website of microPython:
http://micropython.org/

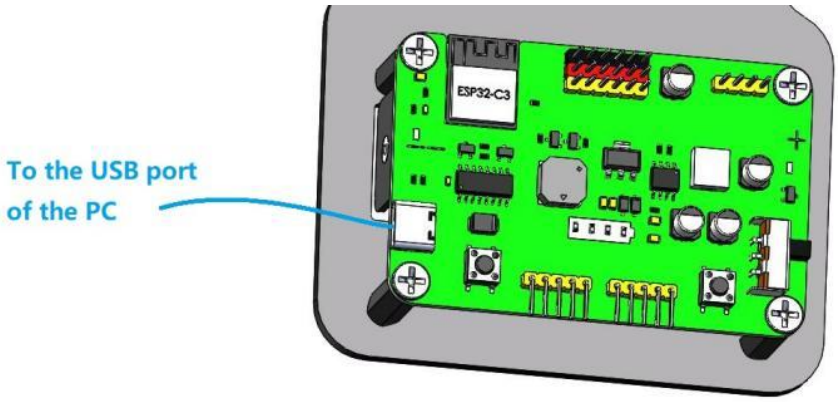
Webpage listing firmware of microPython for ESP32-C3:
https://micropython.org/download/ESP32_GENERIC_C3/

Firmware used in this tutorial is ESP32_GENERIC_C3-20251209-v1.27.0.bin

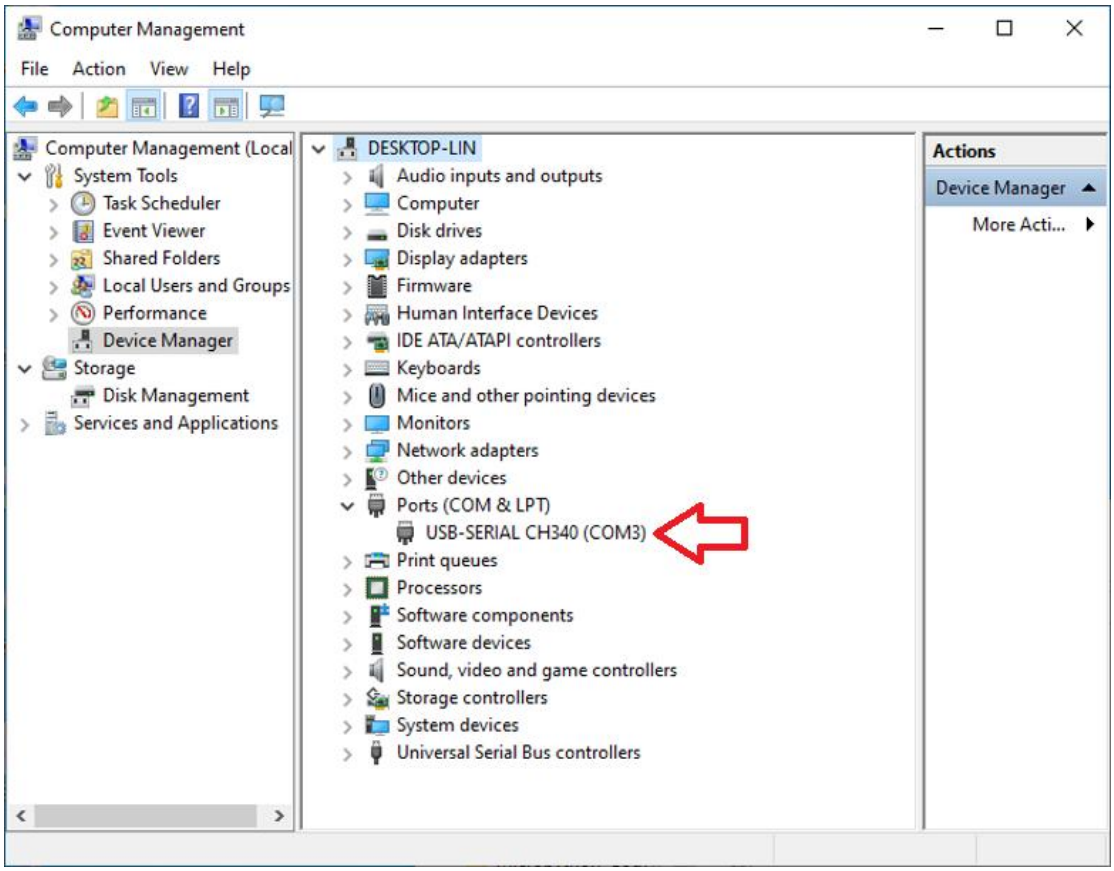
This file is also provided in our tutorial folder : "MicroPython > Firmware".

4.4.2 Burn MicroPython Firmware

Use the USB cable to connect the PC and eArm, as shown below:

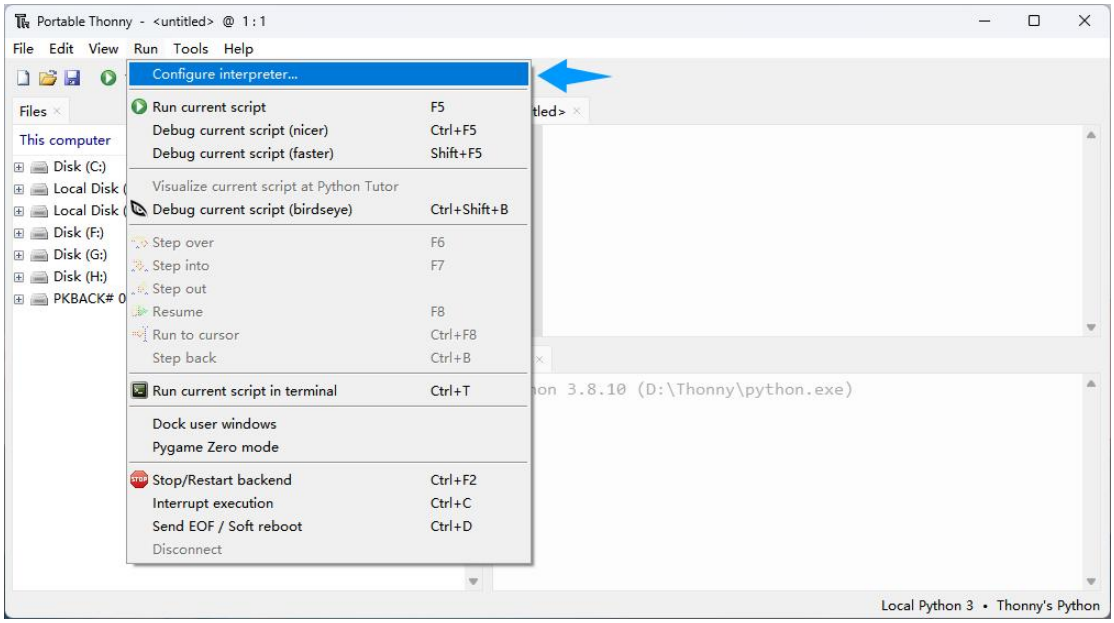


Make sure that the driver has been installed successfully and that it can recognize COM port correctly. Open device manager and expand "Ports".

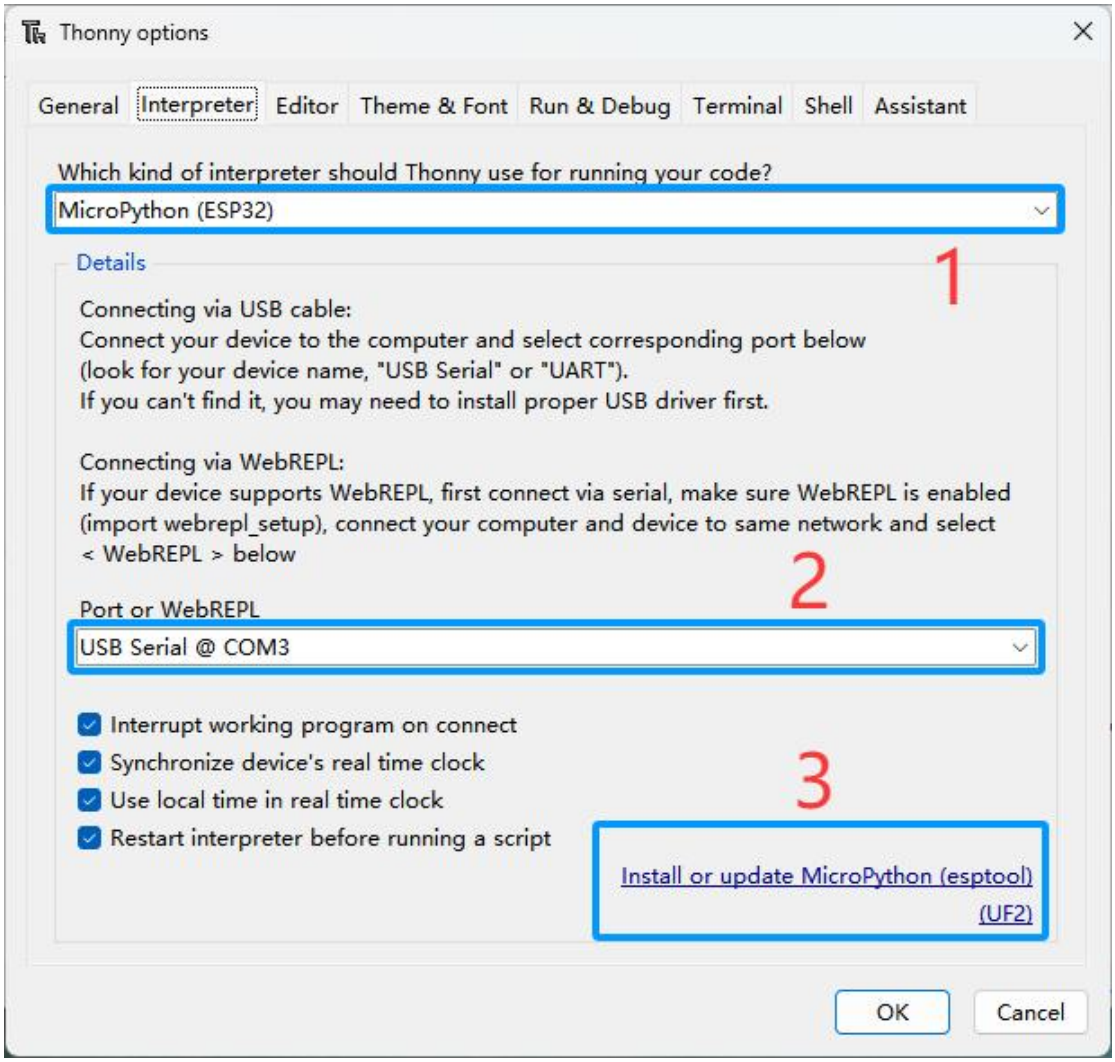


Note: Driver display symbol is "USB-SERIAL CH340 (COMx)", "x" can be any number, it depends on what number your computer assigns to the ESP32 device.

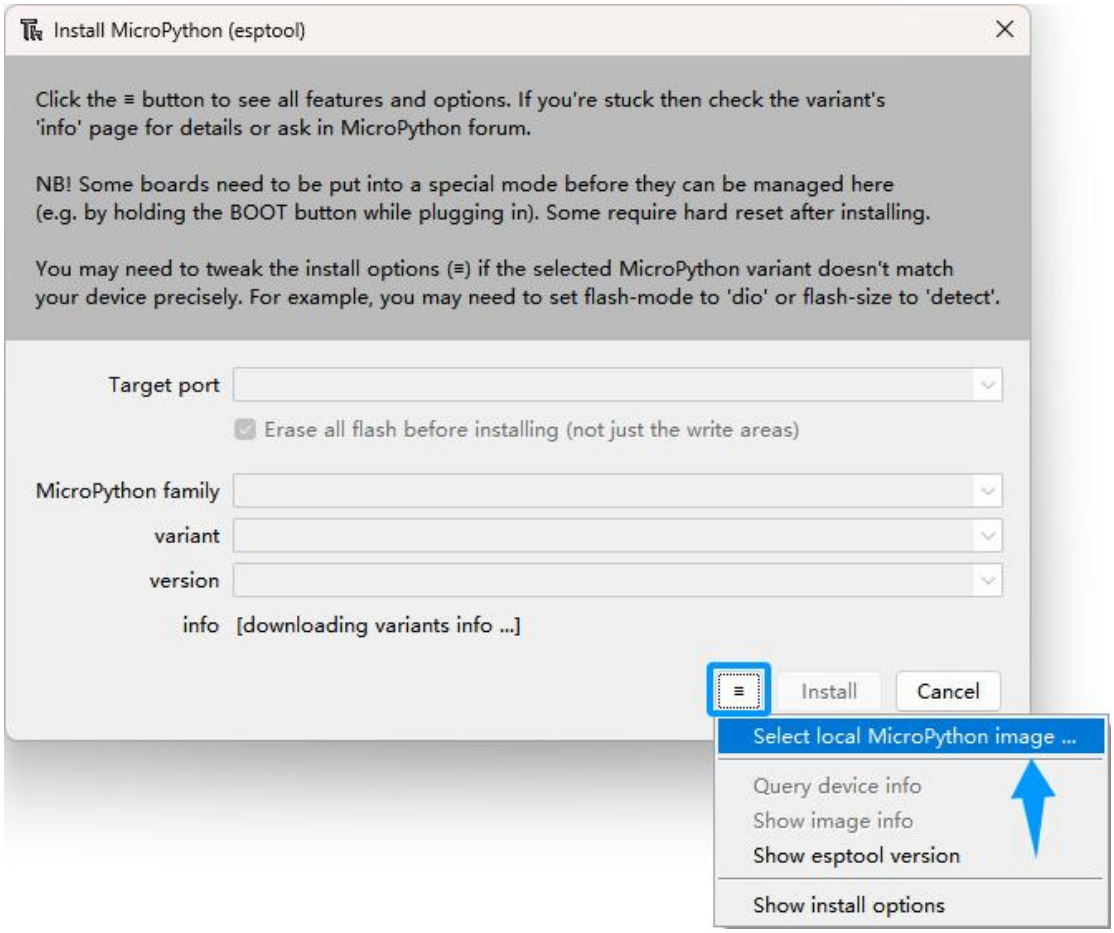
1. Open Thonny, click “Run” and select “Configure interpreter...”



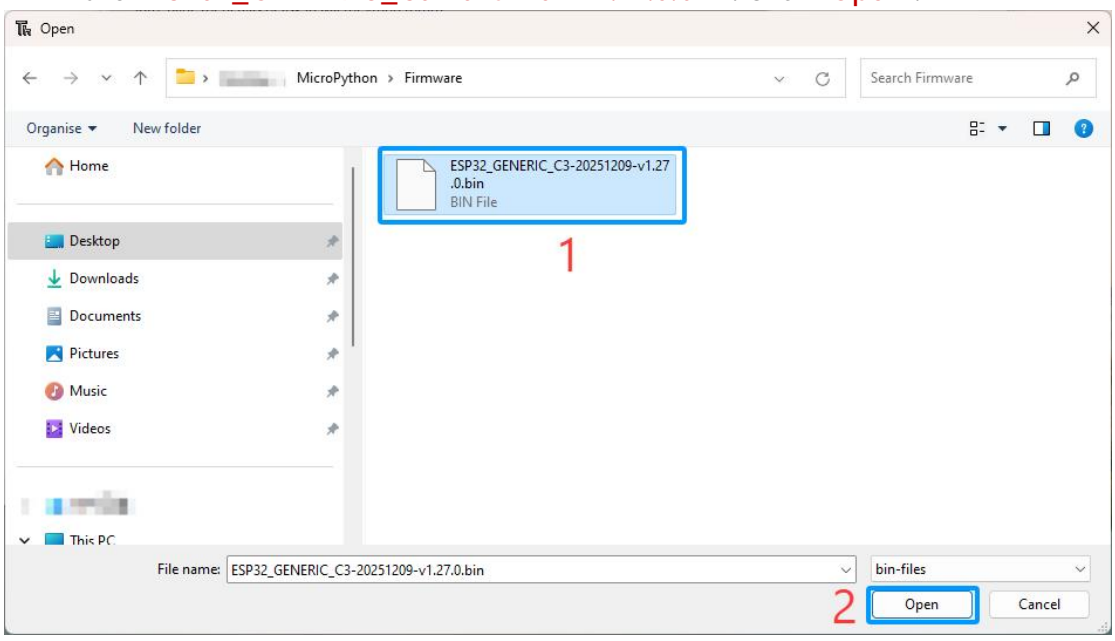
2. Select “Micropython (ESP32)”, select “USB Serial @ (COM3)”, and then click the long button under “Install or update MicroPython(esptool)(UF2)”.



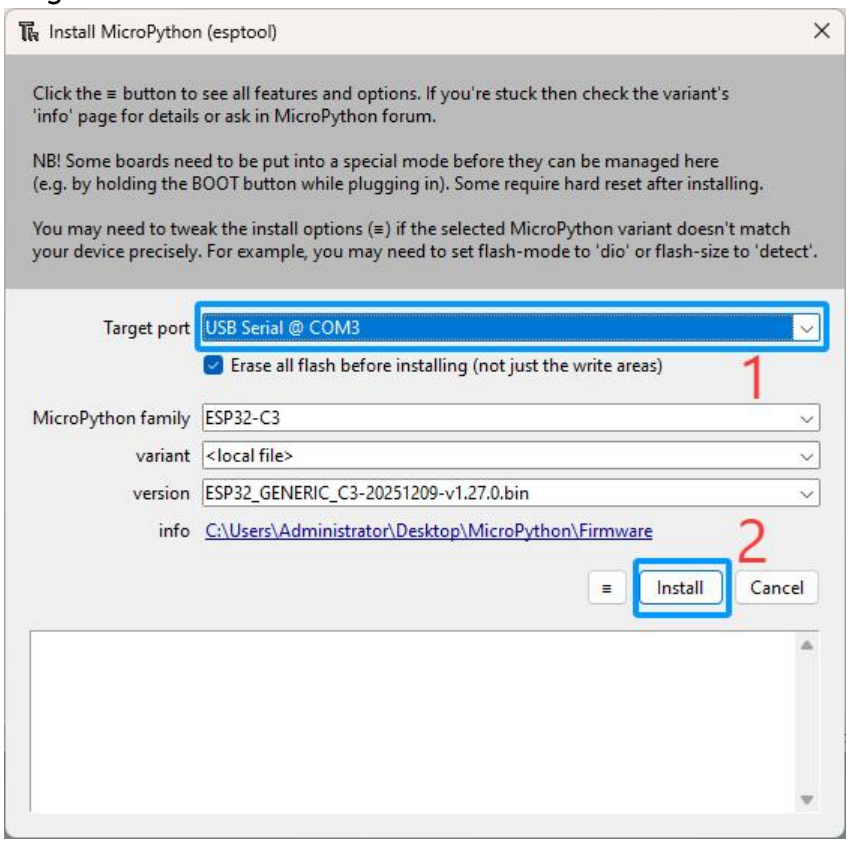
3. The following dialog box pops up. Select “Select local MicroPython image...”.



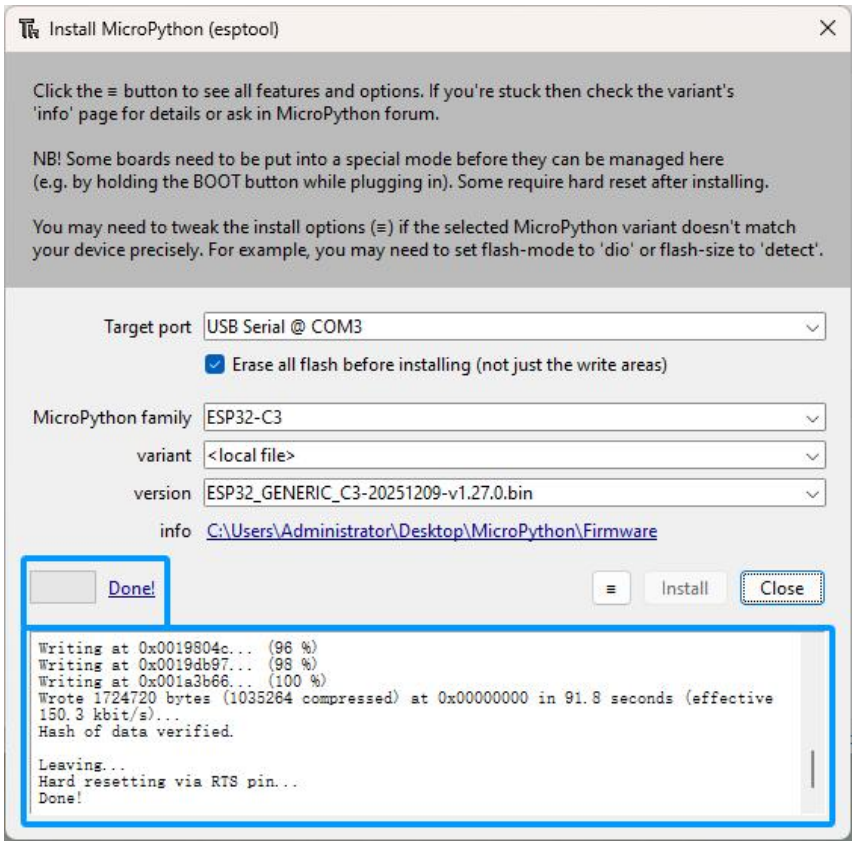
4. The following dialog box pops up. Select the previous prepared microPython firmware “ESP32_GENERIC_C3-20251209-v1.27.0.bin”. Click "Open".



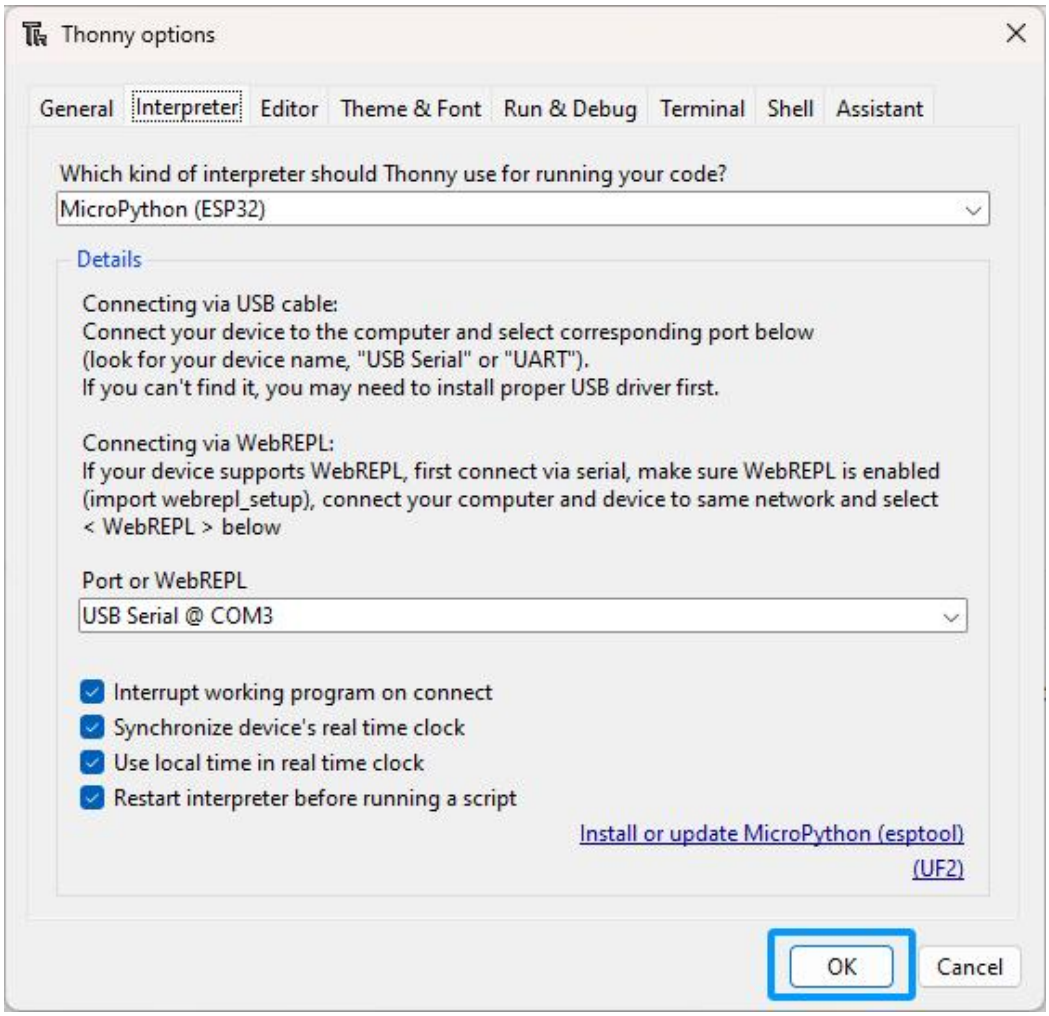
5. Select “Micropython (ESP32)”, select “USB Serial @ (COM3)”, and then click the long button under “Install”.



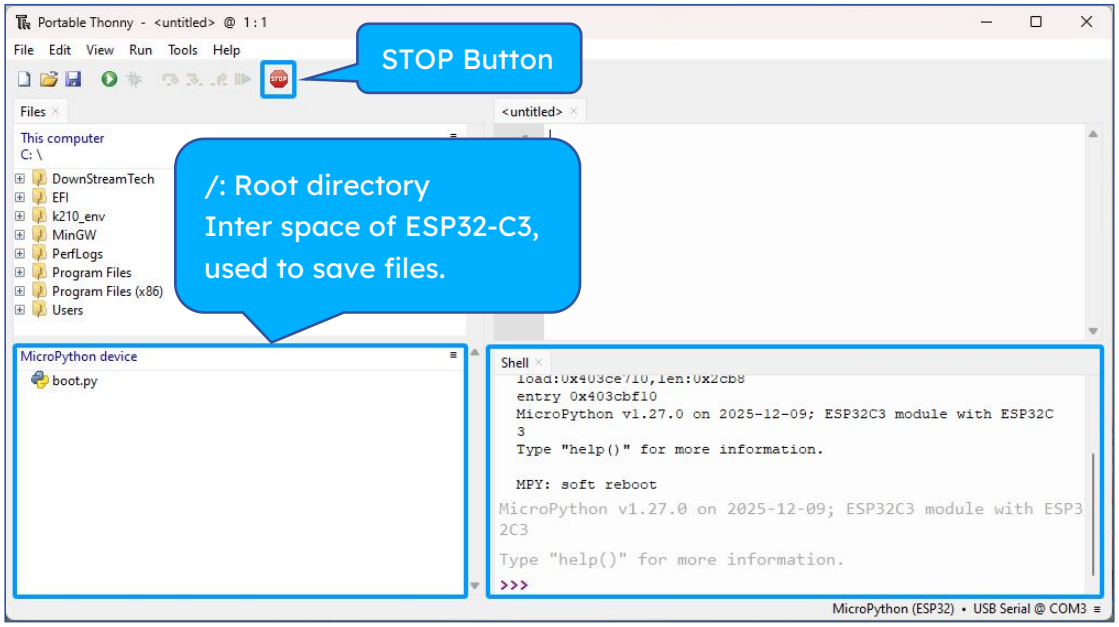
6. Wait for the installation to be done, and then click "Close".



7. Click "OK".



8. Close all dialog boxes, turn to main interface and click “STOP”. As shown in the illustration below:

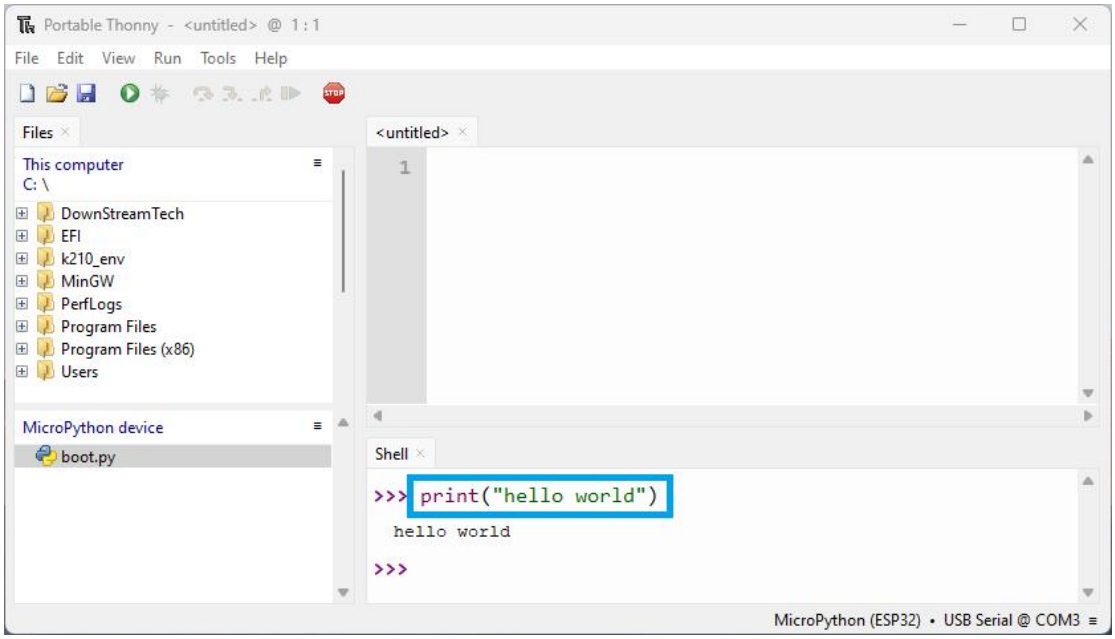


9. So far, all the preparations have been made.

4.5 Basic Usage of Thonny

4.5.1 Test Shell commands

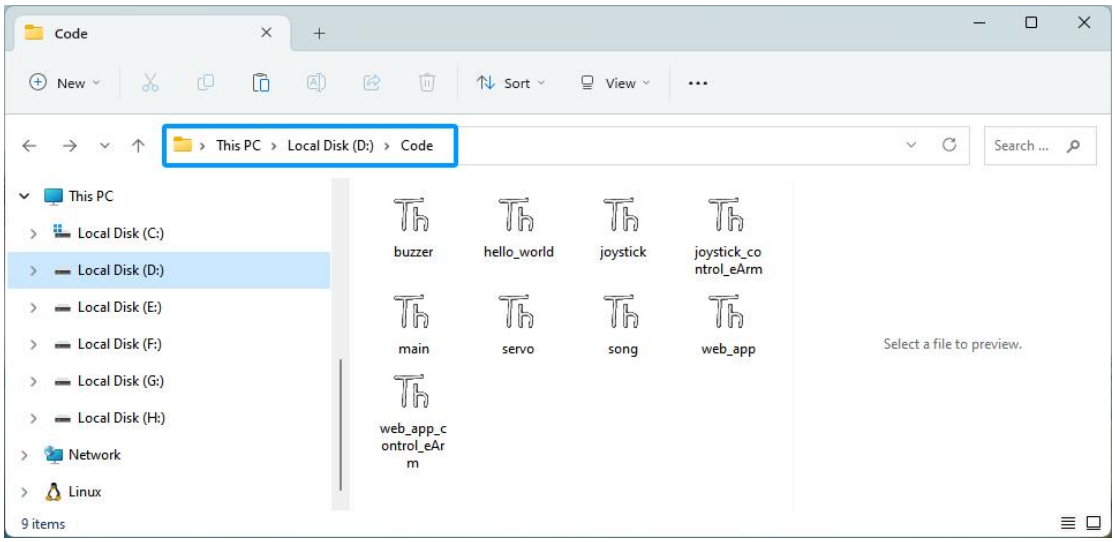
Enter “print("hello world")” in “Shell” and press Enter.



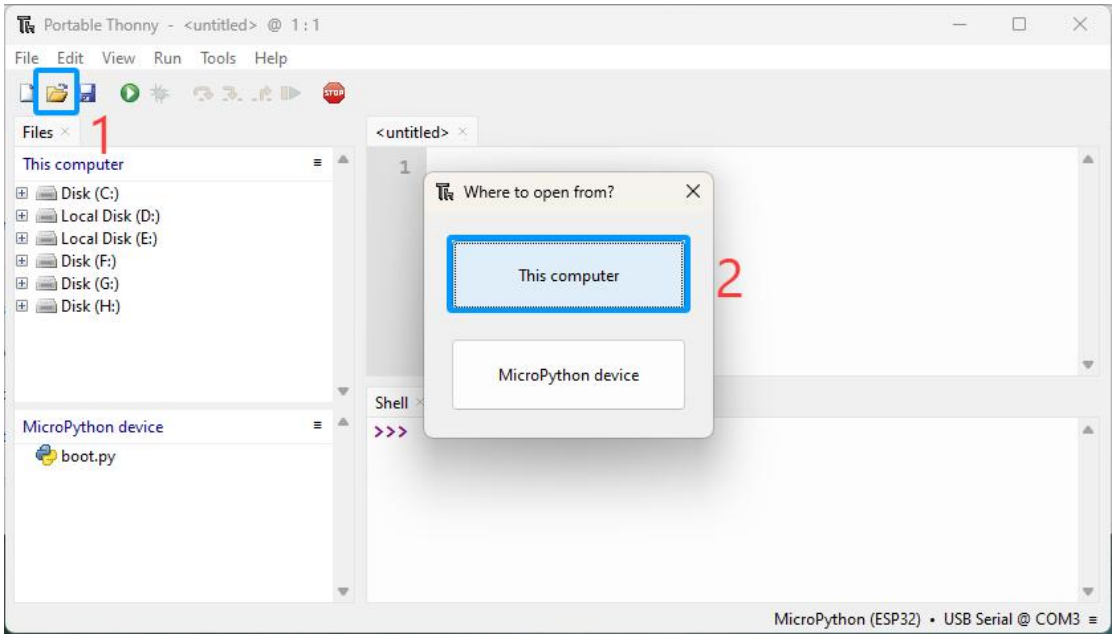
4.5.2 Run Online

ESP32 needs to be connected to a computer when it is run online. Users can use Thonny to writer and debug programs.

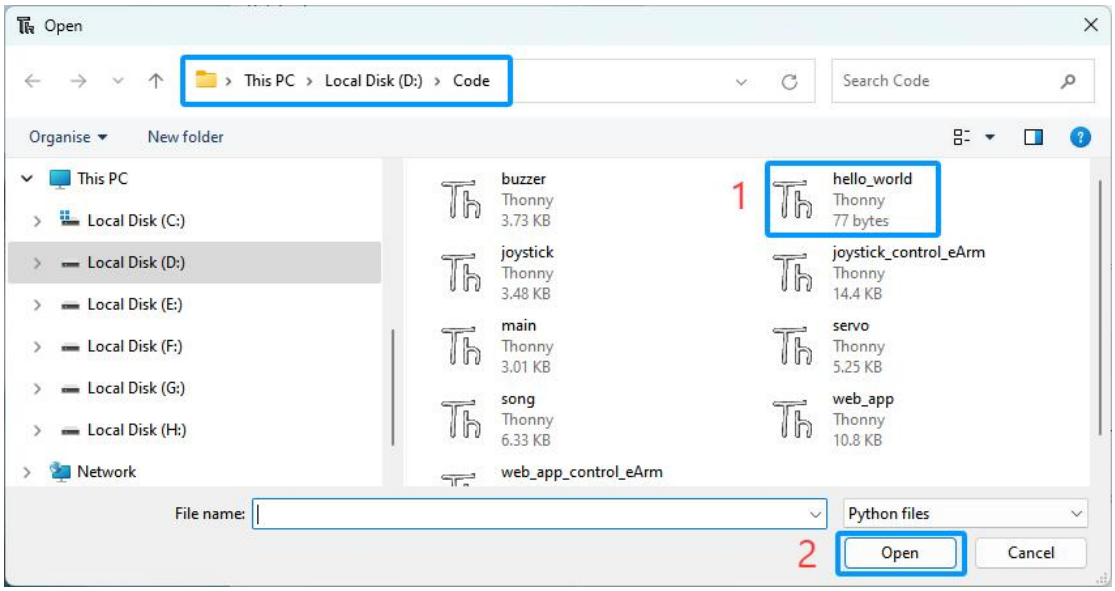
Advance the program folder “... /MicroPython/Code” Move to disk (D).



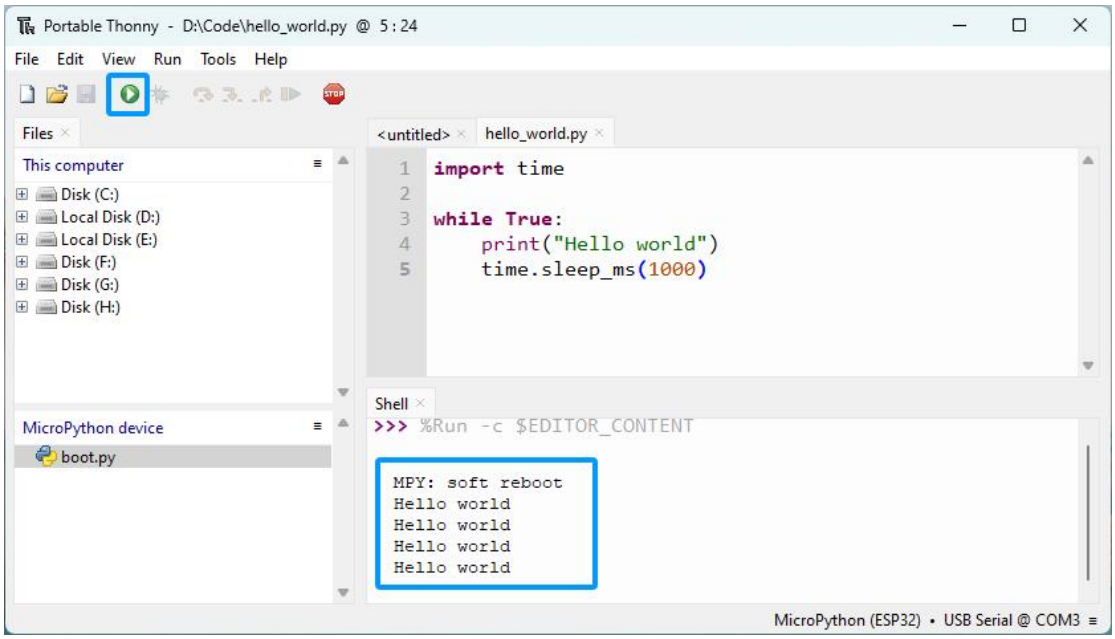
Open“Thonny”, click “Open...”, then click “This computer”.



In the new dialog box, select “ hello_world.py ” in “ D:\Code” folder.



Click “Run current script” to execute the program and “hello world” will be printed in “Shell”.



4.5.3 Run Offline

After ESP32 is reset, it runs the file boot.py in root directory first and then runs file main.py, and finally, it enters “Shell”. Therefore, to make ESP32 execute user’s programs after resetting, we need to add a guiding program in main.py to execute user’s code.

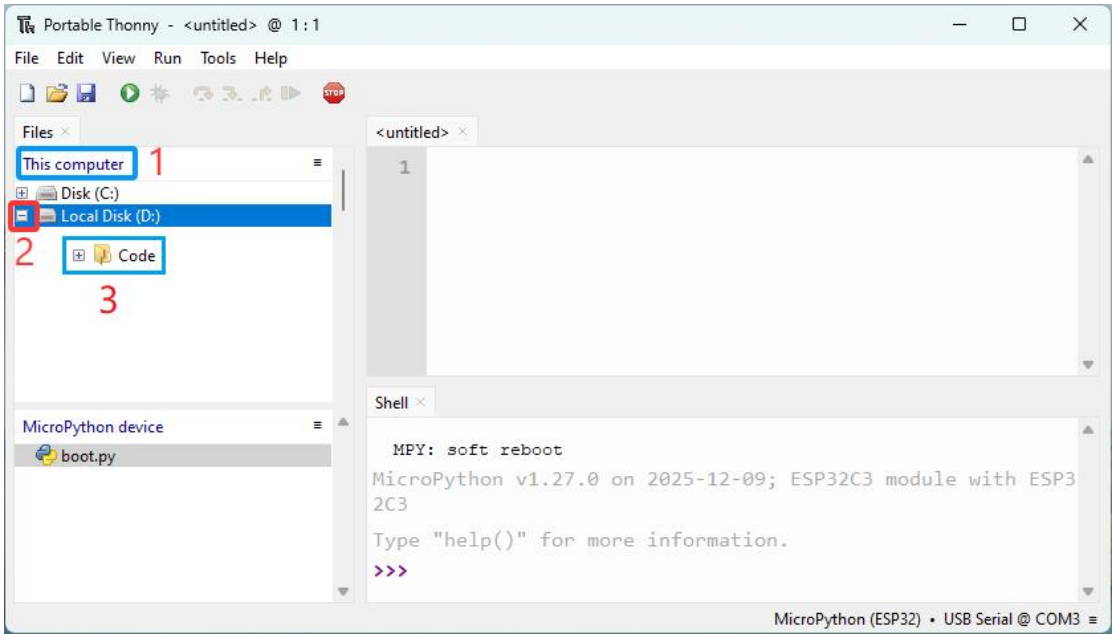
main.py

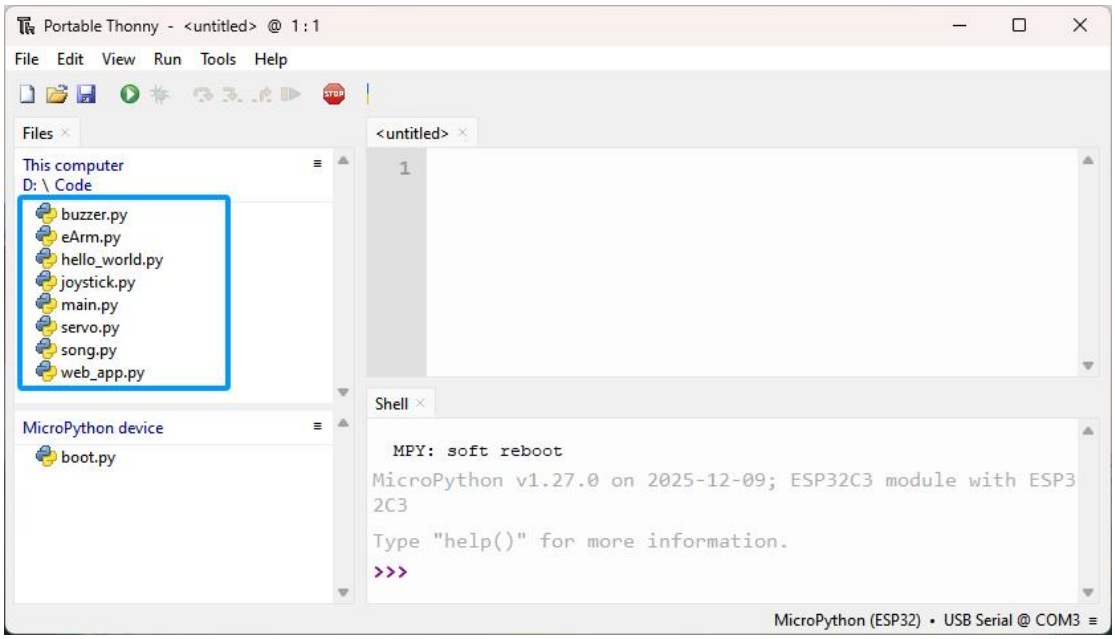
```
"""
This is a Python script executor that will:
    1. Scan all files in the current directory
    2. Skip the boot.py and main.py files
    3. Only execute files with .py extensions
    4. Ensure only files are executed (not directories)
"""
#!/opt/bin/lv_micropython
# Import MicroPython specific modules
import uos as os                # MicroPython's os module
import uerrno as errno          # MicroPython's errno module
```

```
# Get directory iterator for listing files in current directory
# os.ilistdir() returns an iterator of directory entries
iter = os.ilistdir()

# File type constants for directory entries
# These constants come from stat
# module but MicroPython uses different approach
IS_DIR = 0x4000         # Flag indicating the entry is a directory
IS_REGULAR = 0x8000   # Flag indicating the entry is a regular file

# Main loop: Process each entry in the directory
while True:
    try:
        # Get next directory entry using iterator
        # Each entry is a tuple: (name, type, inode[, size])
        entry = next(iter)

        # Extract filename (first element of tuple)
        filename = entry[0]

        # Extract file type (second element of tuple)
        # This indicates if it's a file, directory, etc.
        file_type = entry[1]

        # Skip system files that should not be executed
        # boot.py: Runs on boot, should not be executed manually
        # main.py: Main application file, should be run separately
        if filename == 'boot.py' or filename == 'main.py':
            continue   # Skip to next file

        else:
            # Print separator for visual clarity in output
            print("==============================")

            # Print filename without newline (end="")
            print(filename, end="")

            # Check if current entry is a directory
            if file_type == IS_DIR:
                print(", File is a directory")
                print("==============================")
                # Note: Directories are only identified, not processed further

            else:
```

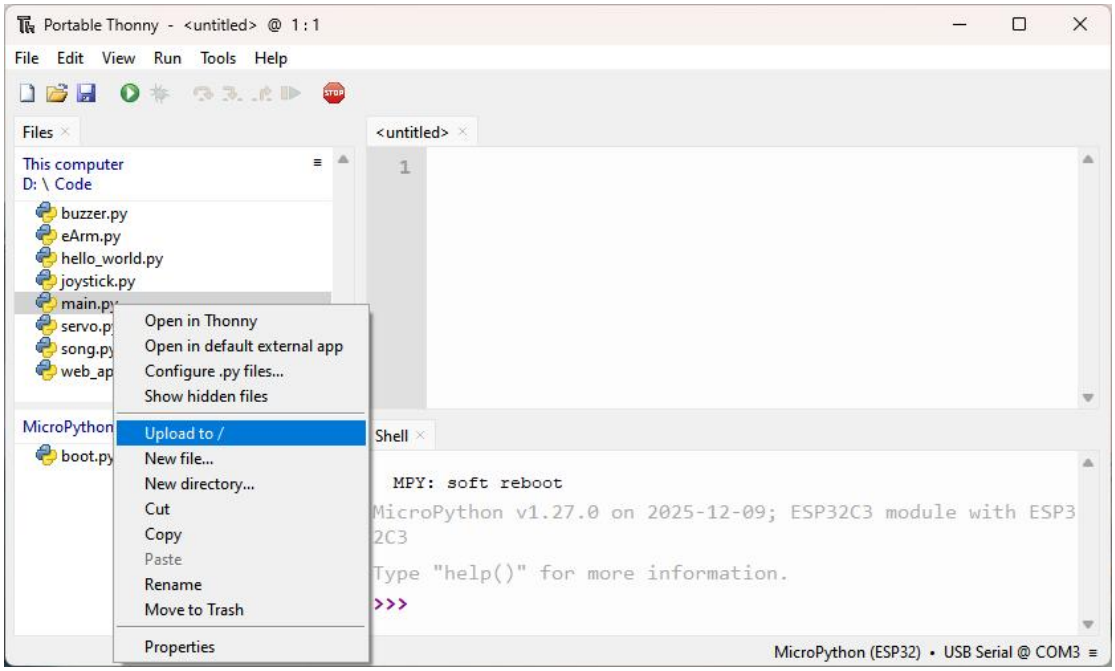
```
                # For regular files, just close the filename line
                print("\n================================")

                # Originally commented code that would print file contents:
                # print("Contents:")
                # with open(filename) as f:
                #       for line in enumerate(f):
                #             print("{}".format(line[1]),end="")
                # print("")

                # Actually execute the Python file
                # exec() runs the code in the file with global context
                # open(filename).read() reads the entire file content
                # globals() provides the current global
                # namespace to the executed code
                exec(open(filename).read(), globals())

    # StopIteration exception is raised when iterator has no more items
    except StopIteration:
        break   # Exit the infinite while loop
```

Expand "Code"folder in the directory of disk(D), and then double-click the "Code" folder to display all the sample codes.



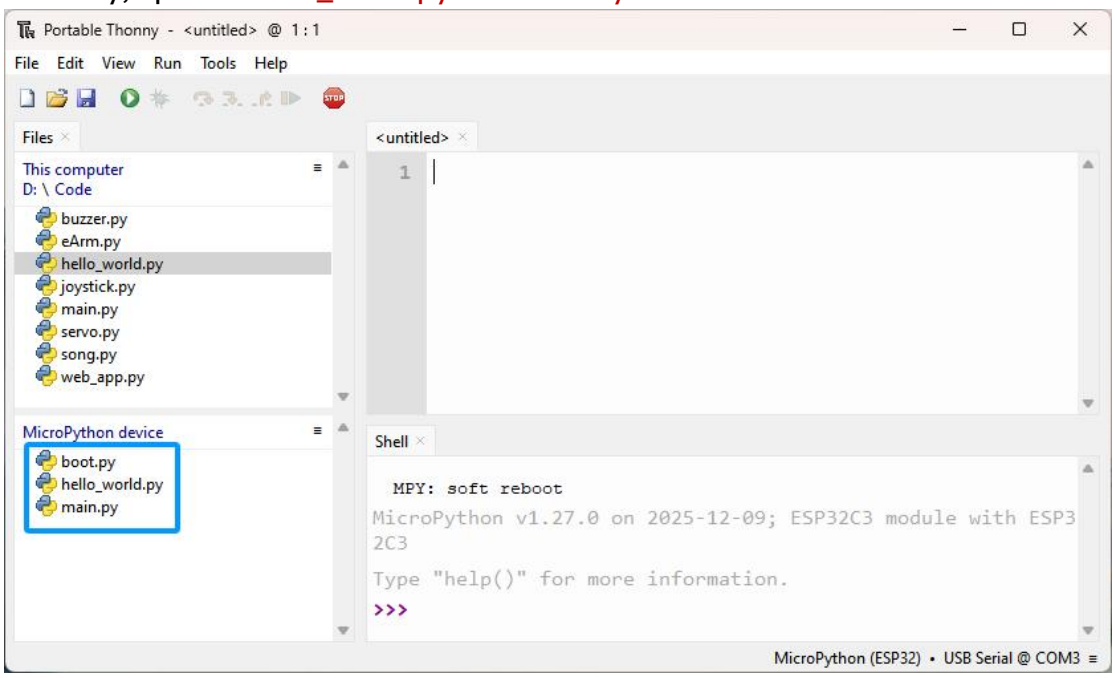


If you want your written programs to run offline, you need to upload main.py we provided and all your codes to "MicroPython device", and then press the reset button on the control board. Here we use programs "hello_world.py" as examples. Select "main.py", right-click to select "Upload to /".

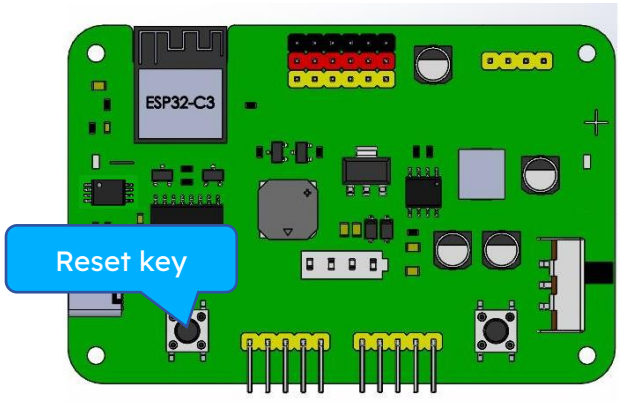


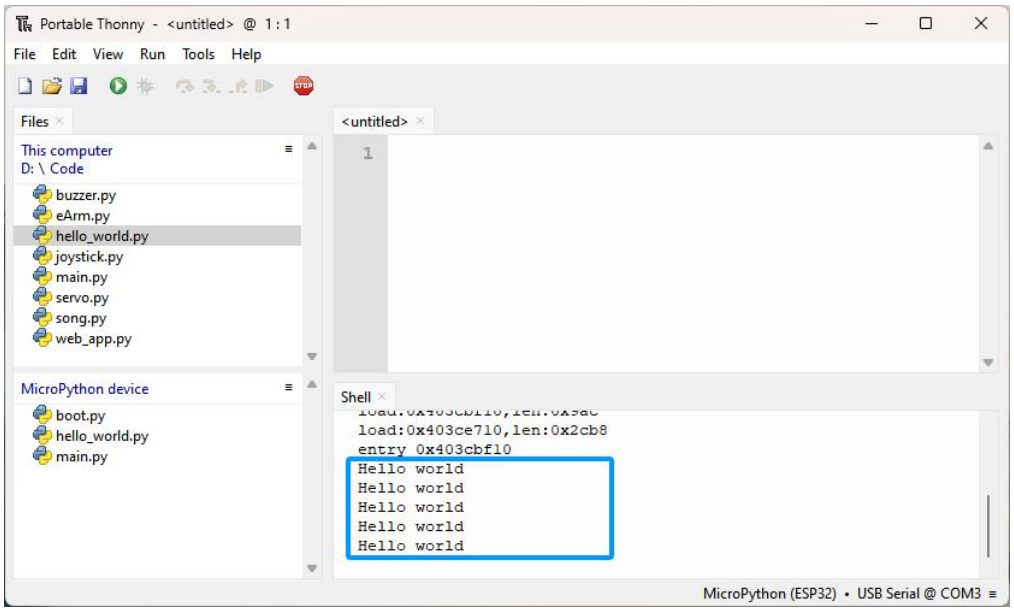
Note: You must stop running online before uploading files to "MicroPython device"! (Click the "STOP" menu in Thonny)

Similarly, upload "hello_world.py" to "MicroPython device".



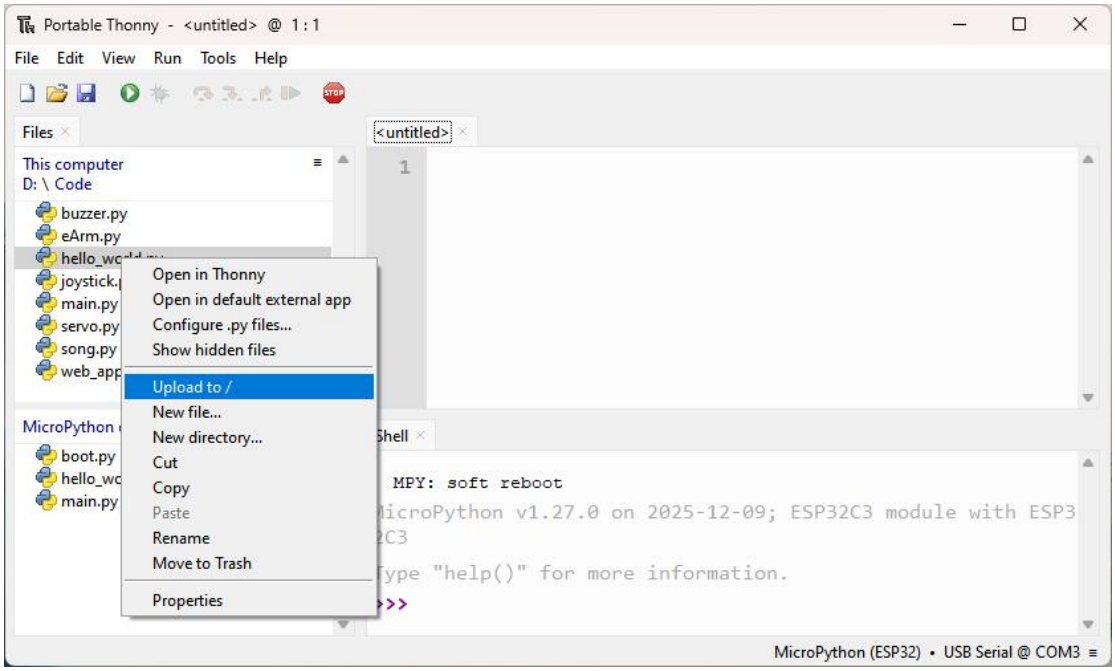
Press the reset key and in the box of the illustration below, you can see the code is executed.



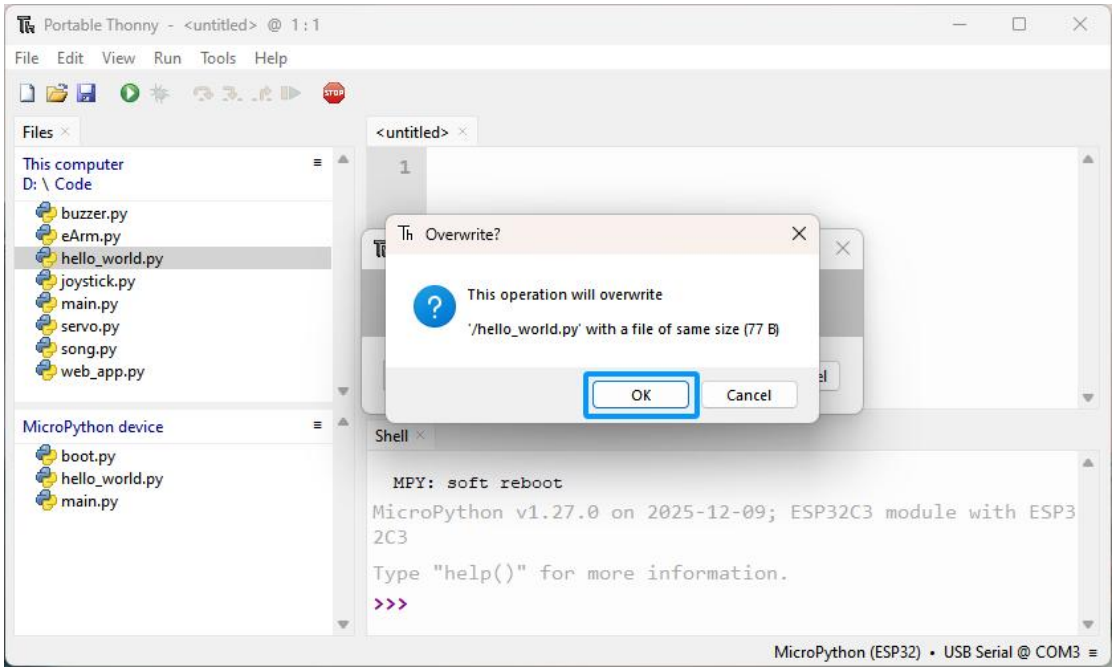


4.5.4 Upload code from computer to ESP32-C3

Select “hello_world.py” , right-click your mouse and select “Upload to /” to upload code to ESP32-C3’s root directory.

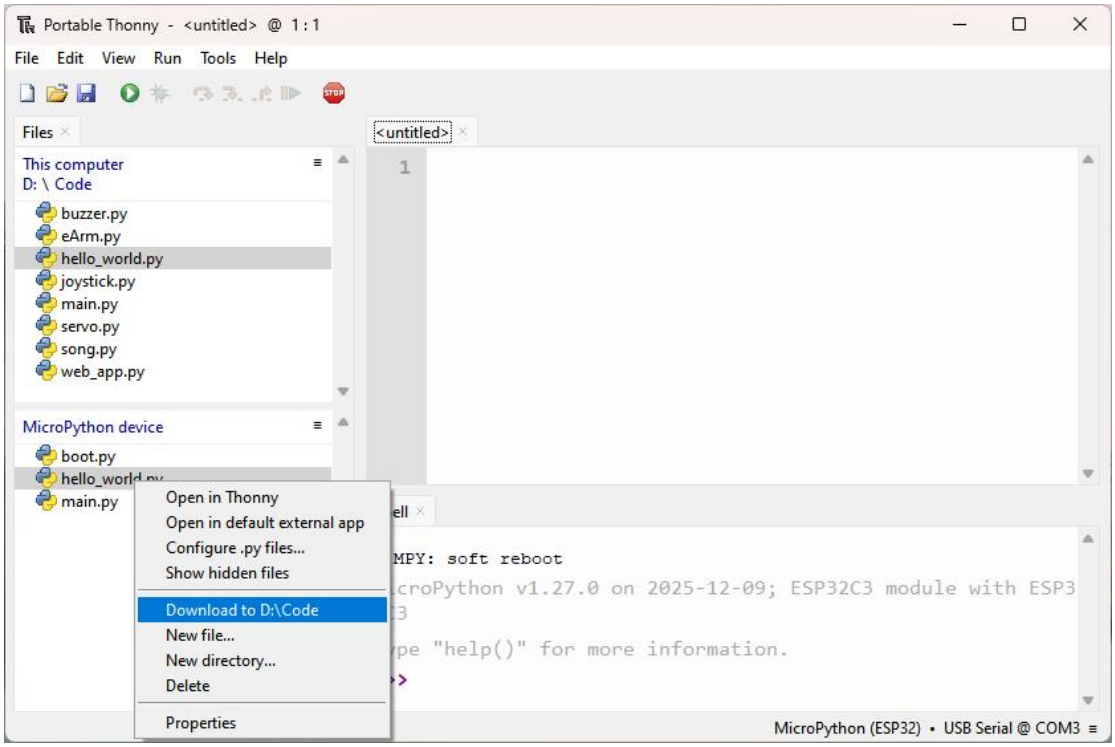


If the pop-up window prompts whether to overwrite the "hello_world.py" file, please click "OK".



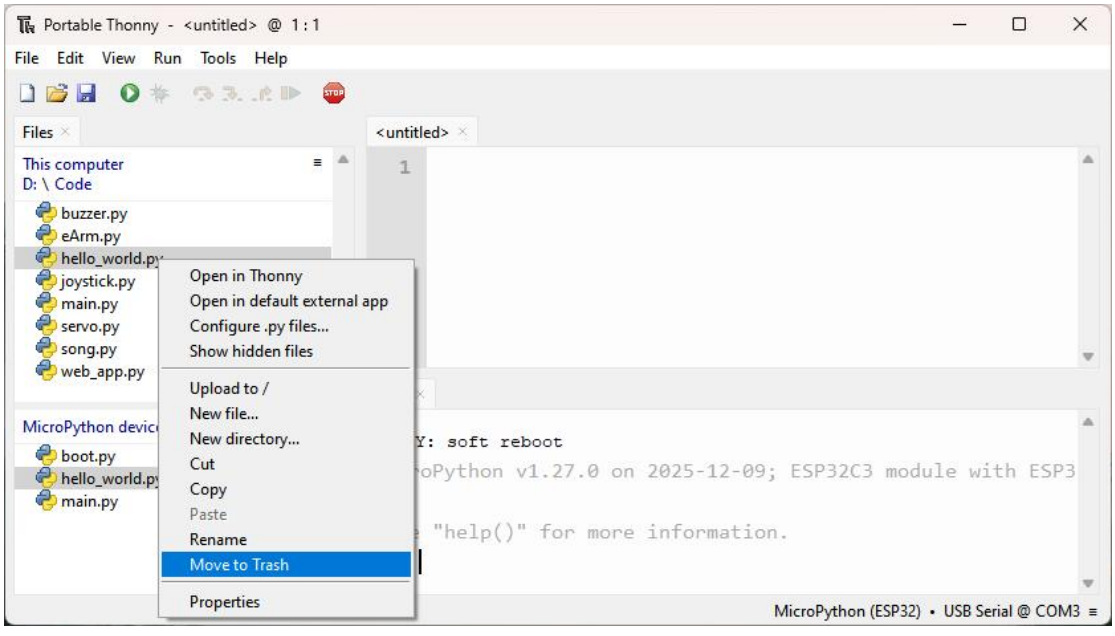
4.5.5 Download the code from ESP32-C3 to computer

Select "hello_world.py" in "MicroPython device", right-click to select "Download to …" to download the code to your computer.



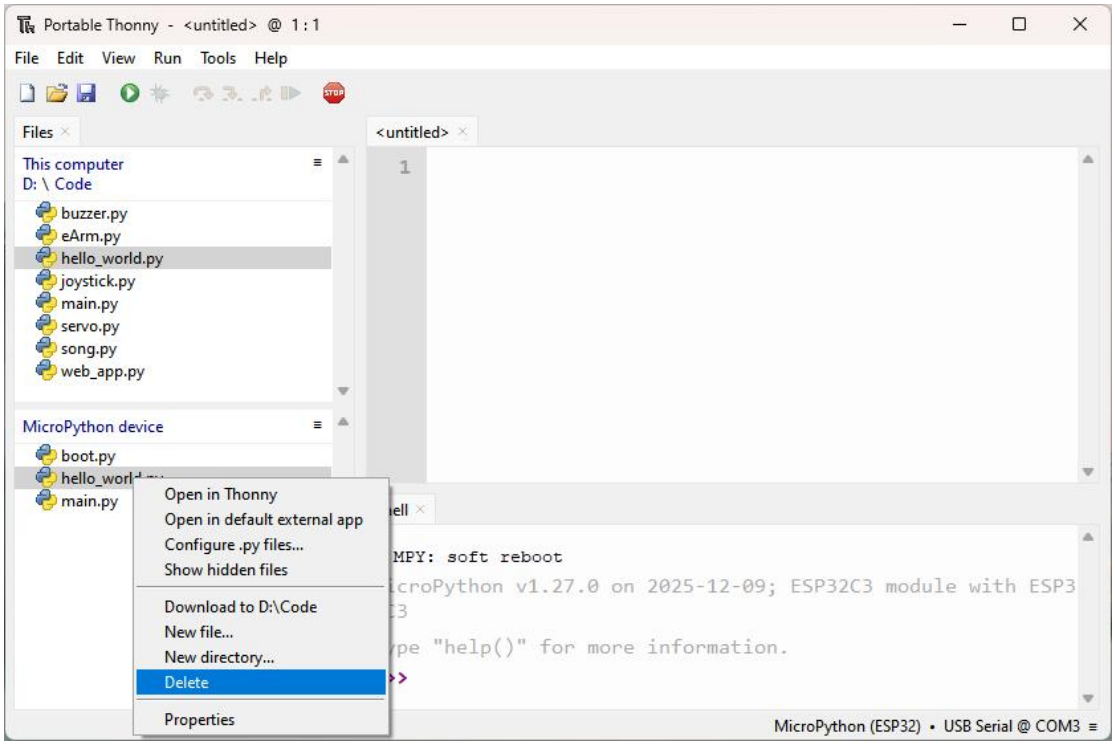
4.5.6 Deleting Files from your Computer Directory

Select "hello_world.py" in "Code", right-click it and select "Move to Recycle Bin" to delete it from "Code".



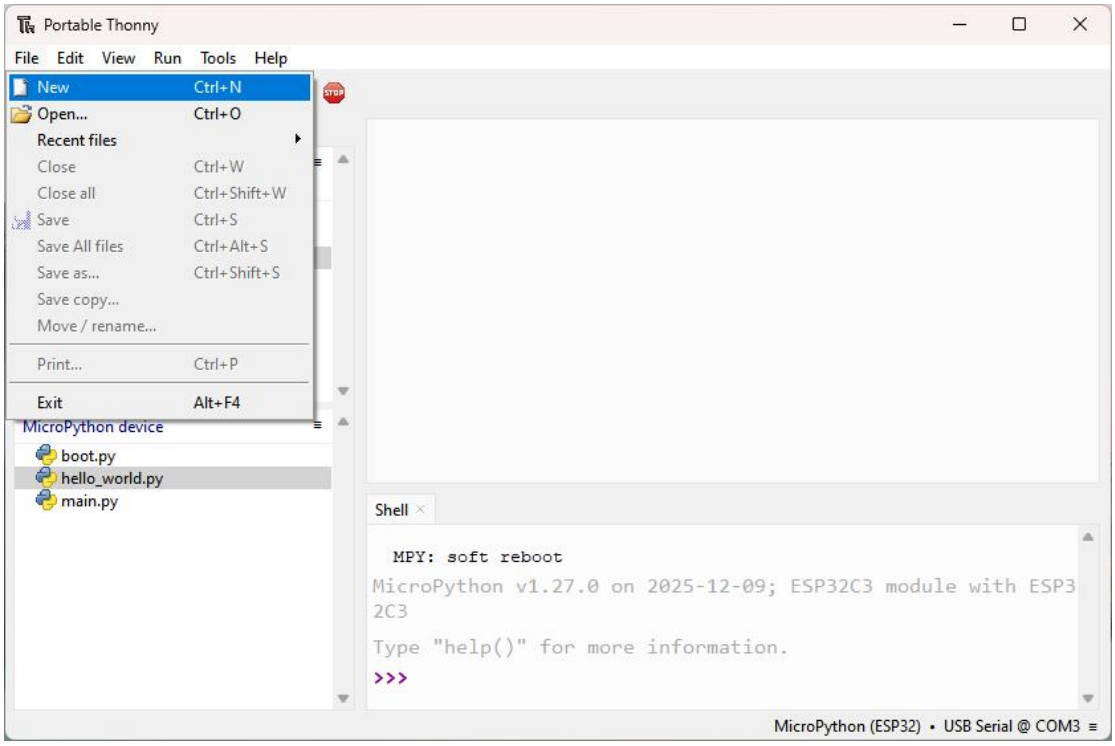
4.5.7 Deleting Files from ESP32-C3’ s Root Directory

Select “hello_world.py” in “MicroPython device”, right-click it and select “Delete” to delete “hello_world.py” from ESP32-C3’s root directory.



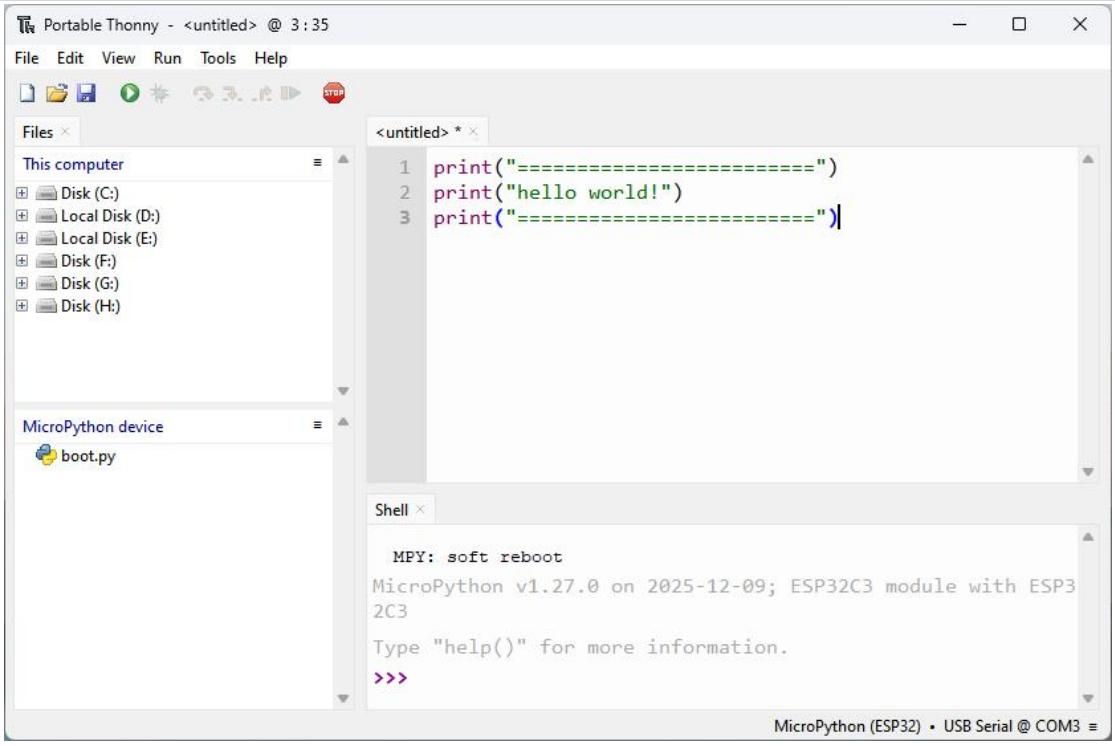
4.5.8 Creating and Saving the code

Click “File” -> “New” to create and write codes.

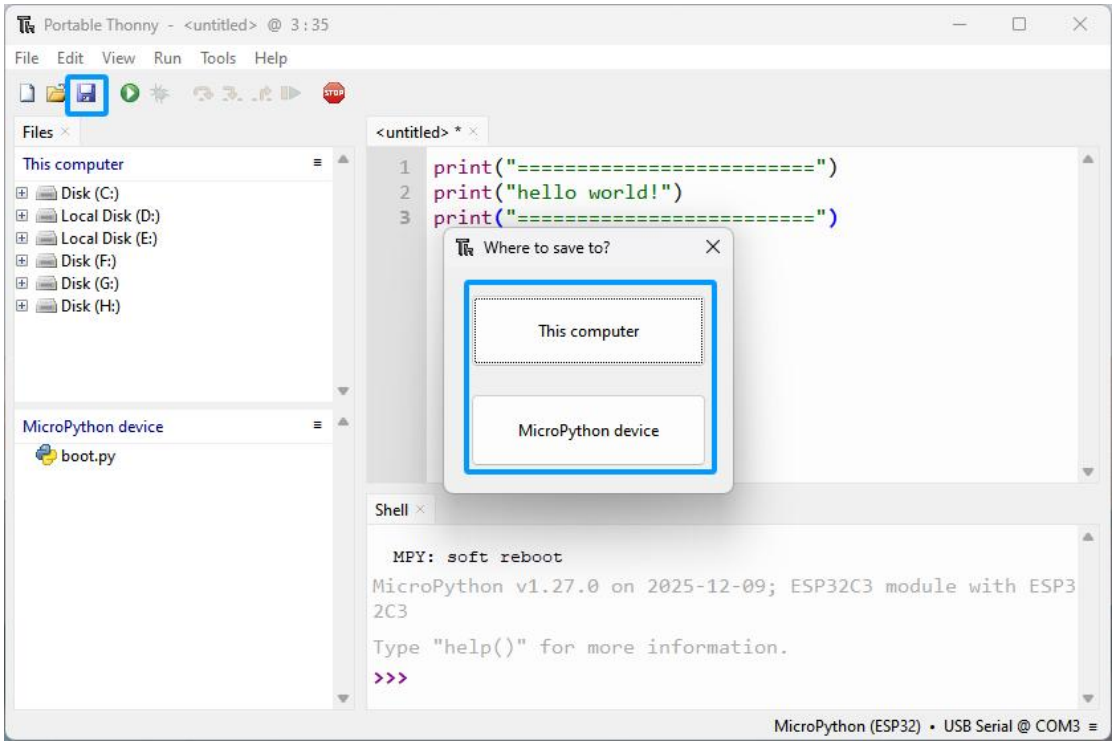


Enter codes in the newly opened file.

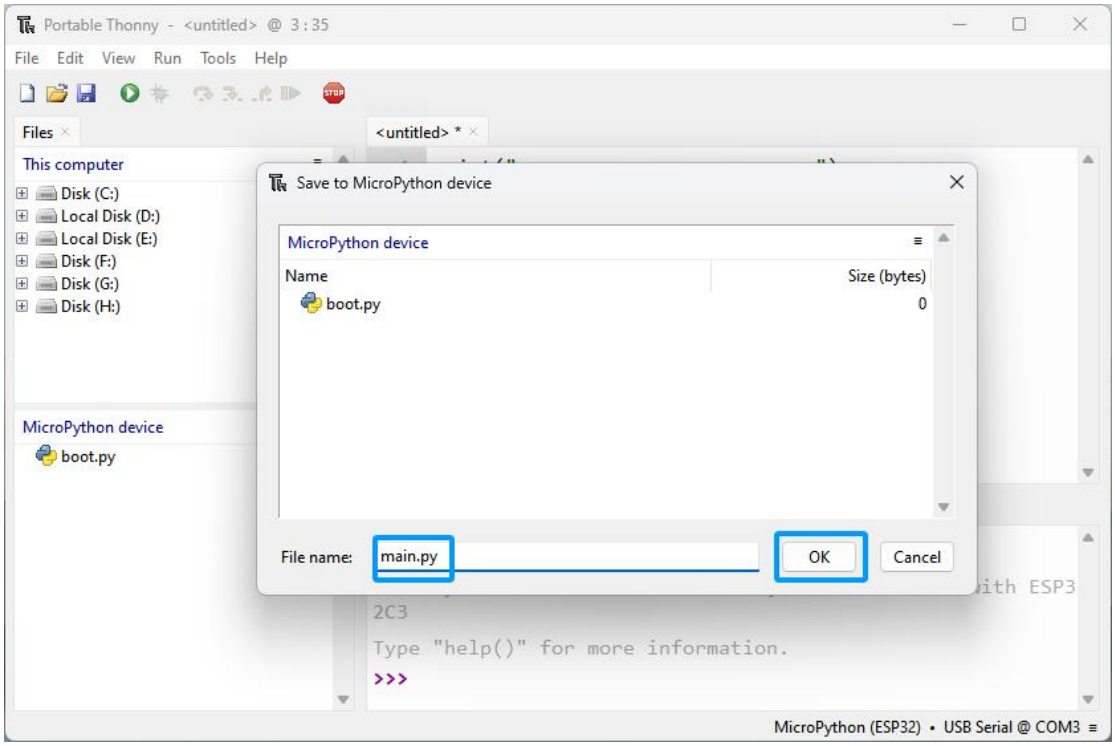
```
print("==========================")
print("hello world!")
print("==========================")
```



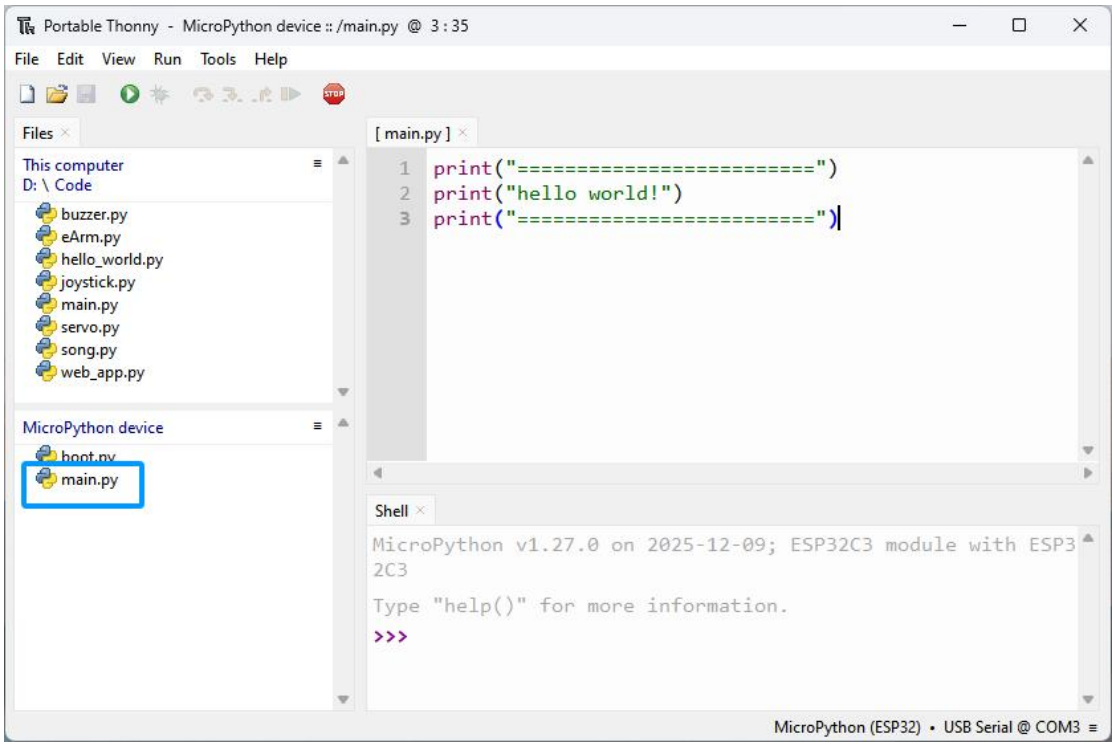
Click “Save” on the menu bar. You can save the codes either to your computer or to ESP32-C3.



Select “MicroPython device”, enter “main.py” in the newly pop-up window and click “OK”.

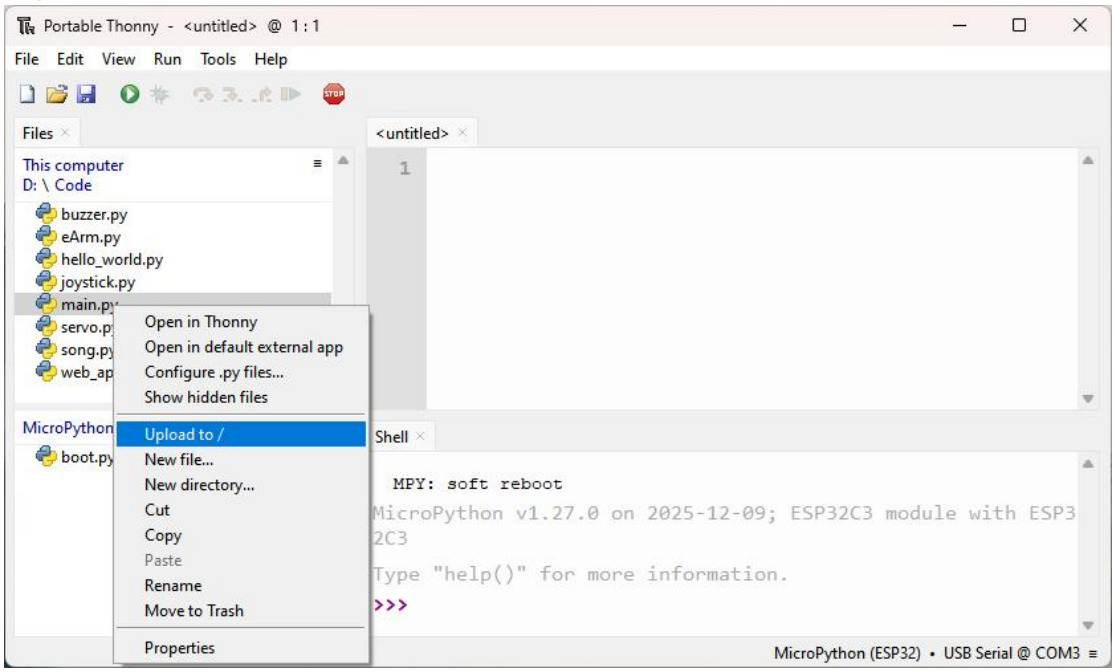


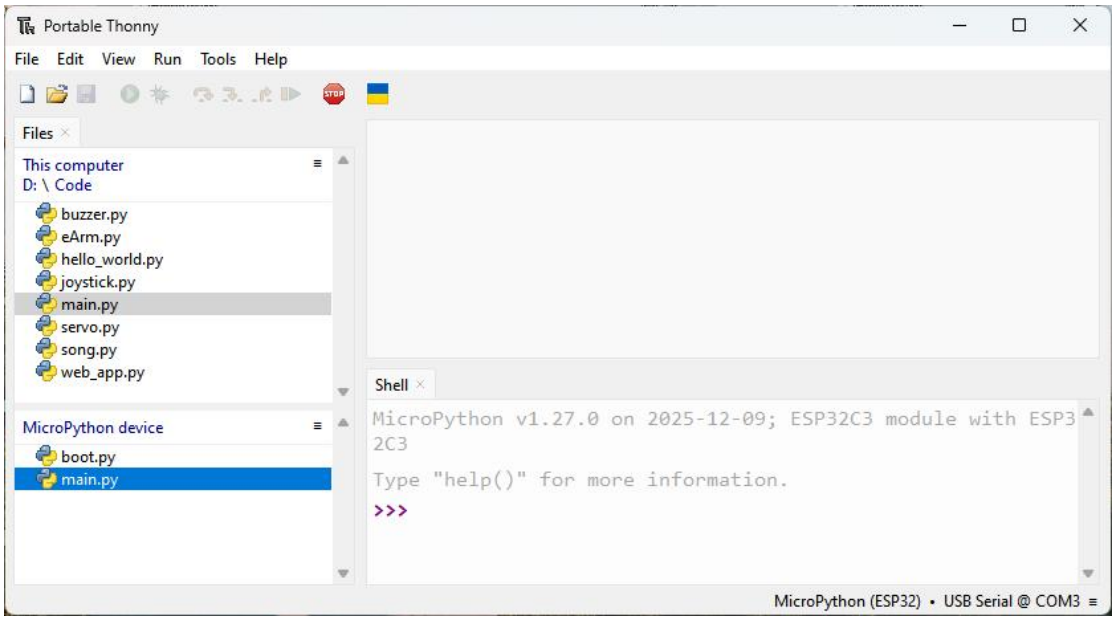
You can see that codes have been uploaded to ESP32-C3.



4.6 Code Examples

To enable the program to run offline, you need to upload the main.py file we provided to the "MicroPython Device". Select "main.py", right-click and select "Upload to/".

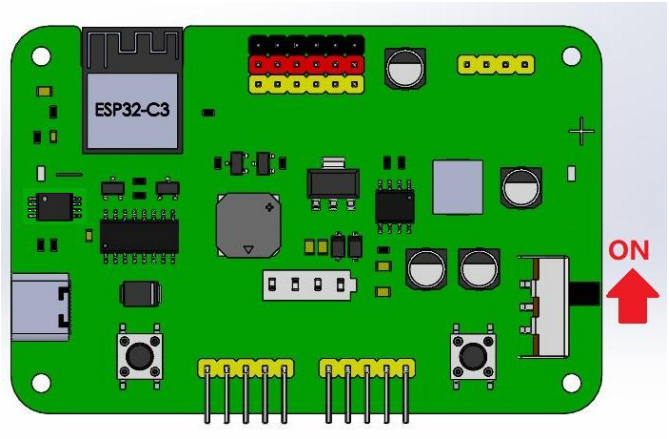


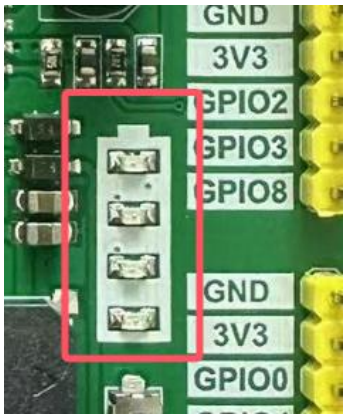


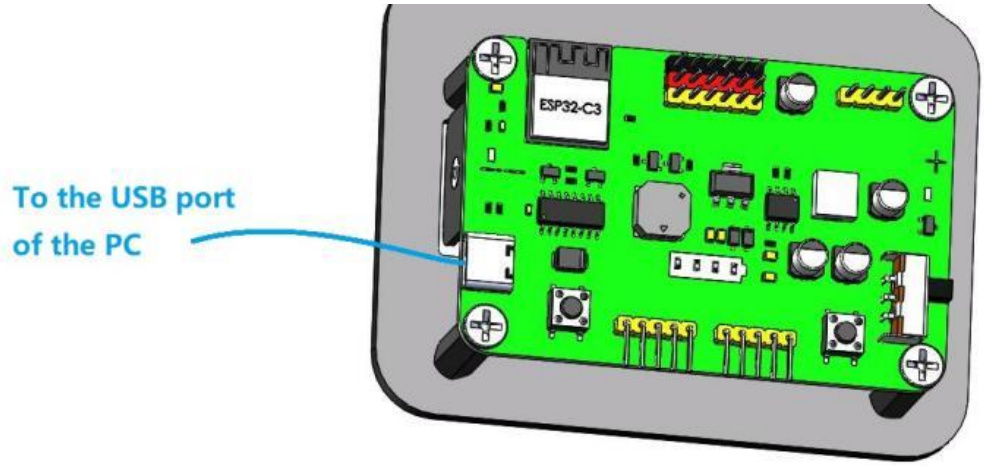
Note: You must stop running online before uploading files to "MicroPython device"! (Click the "STOP" menu in Thonny)

Turn the eArm's power switch to the **ON** position.

■ 1: Make sure the 18650 lithium battery is installed.
■ 2: The battery indicator shows 3 bars or more.

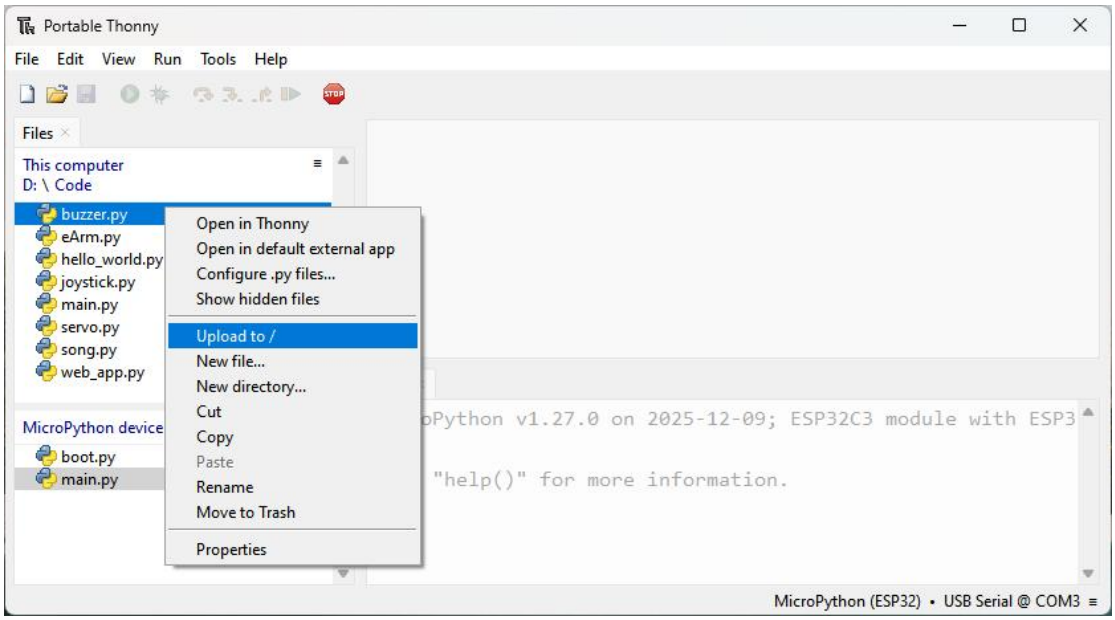


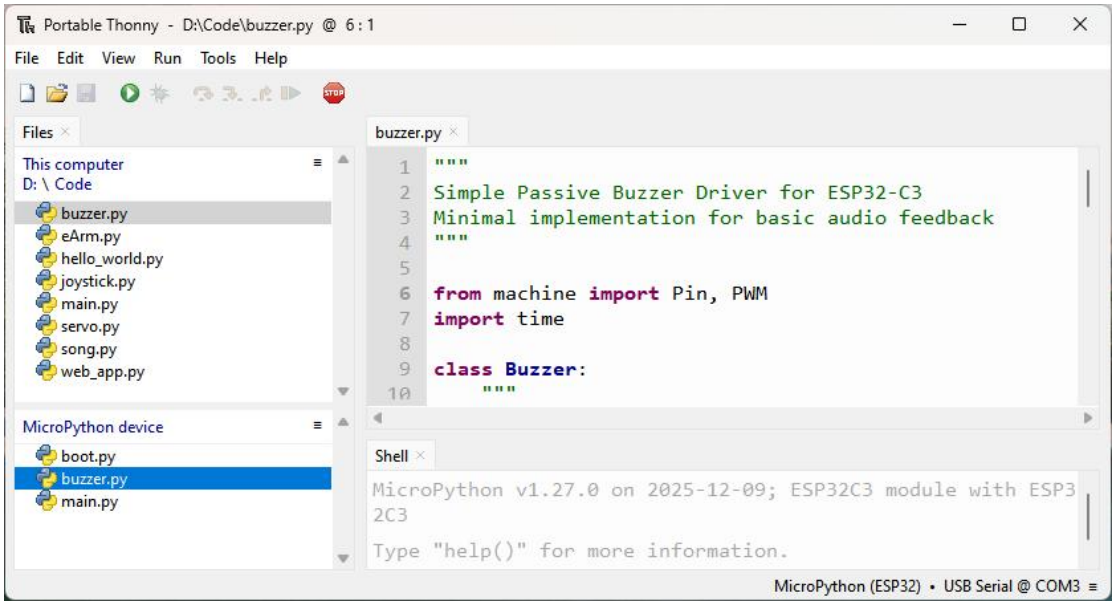




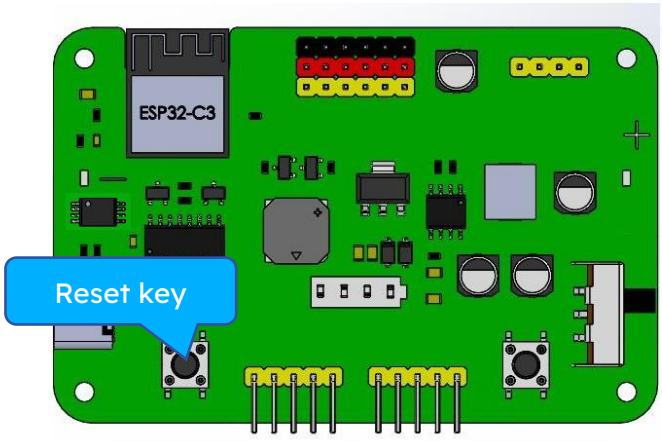
4.5.1 Onboard buzzer

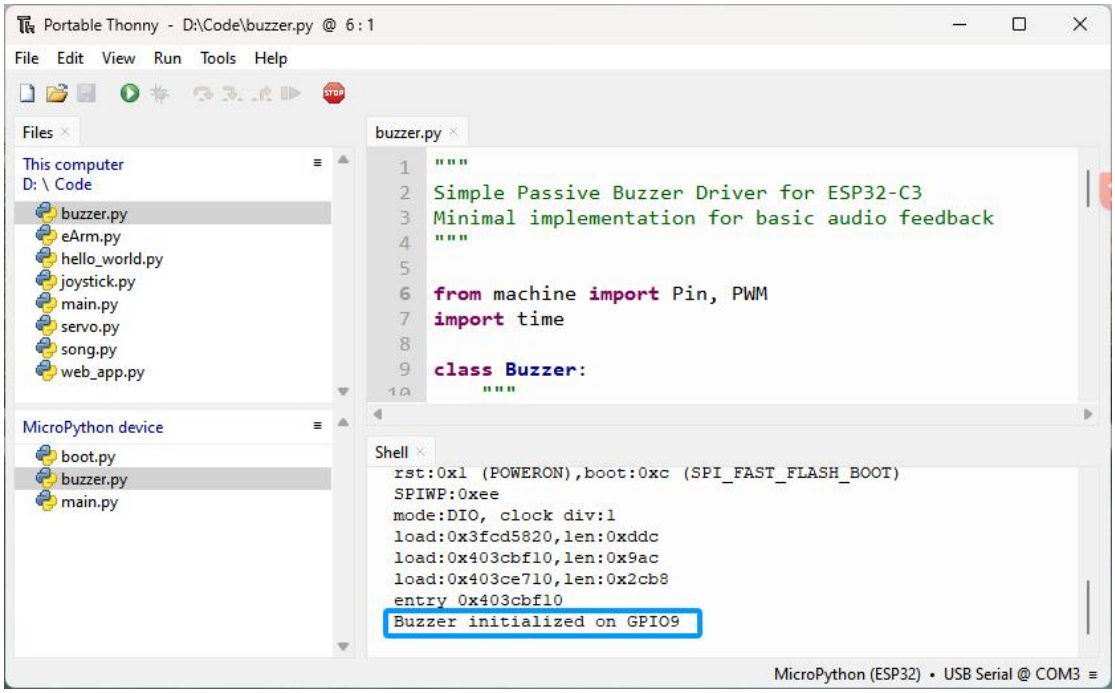
Select "buzzer.py" , right-click your mouse and select "Upload to /" to upload code to ESP32-C3's root directory.



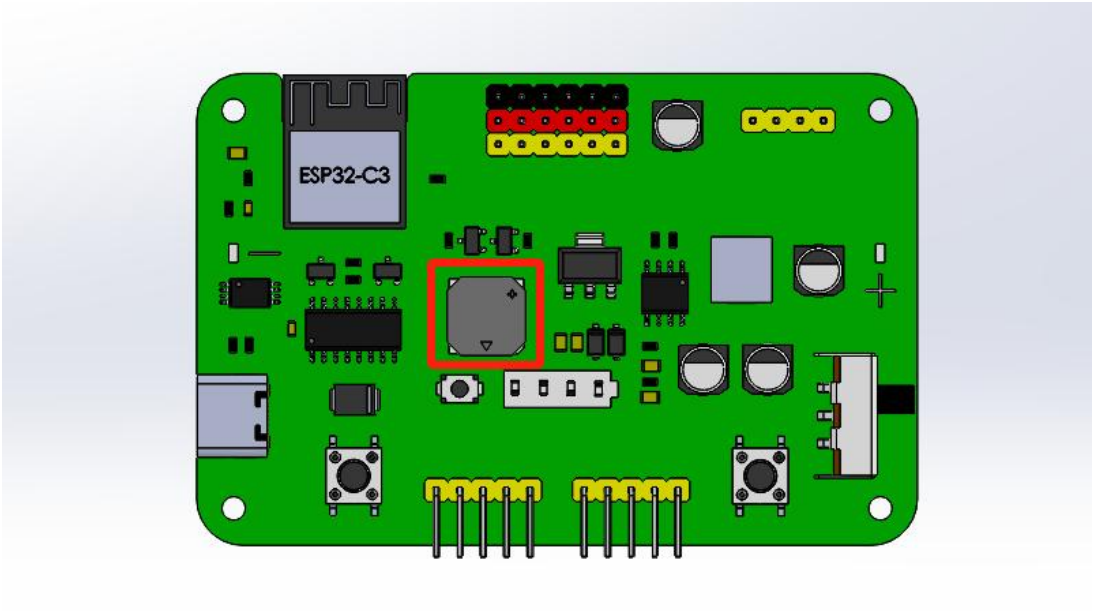


Press the reset key and in the box of the illustration below, you can see the code is executed.



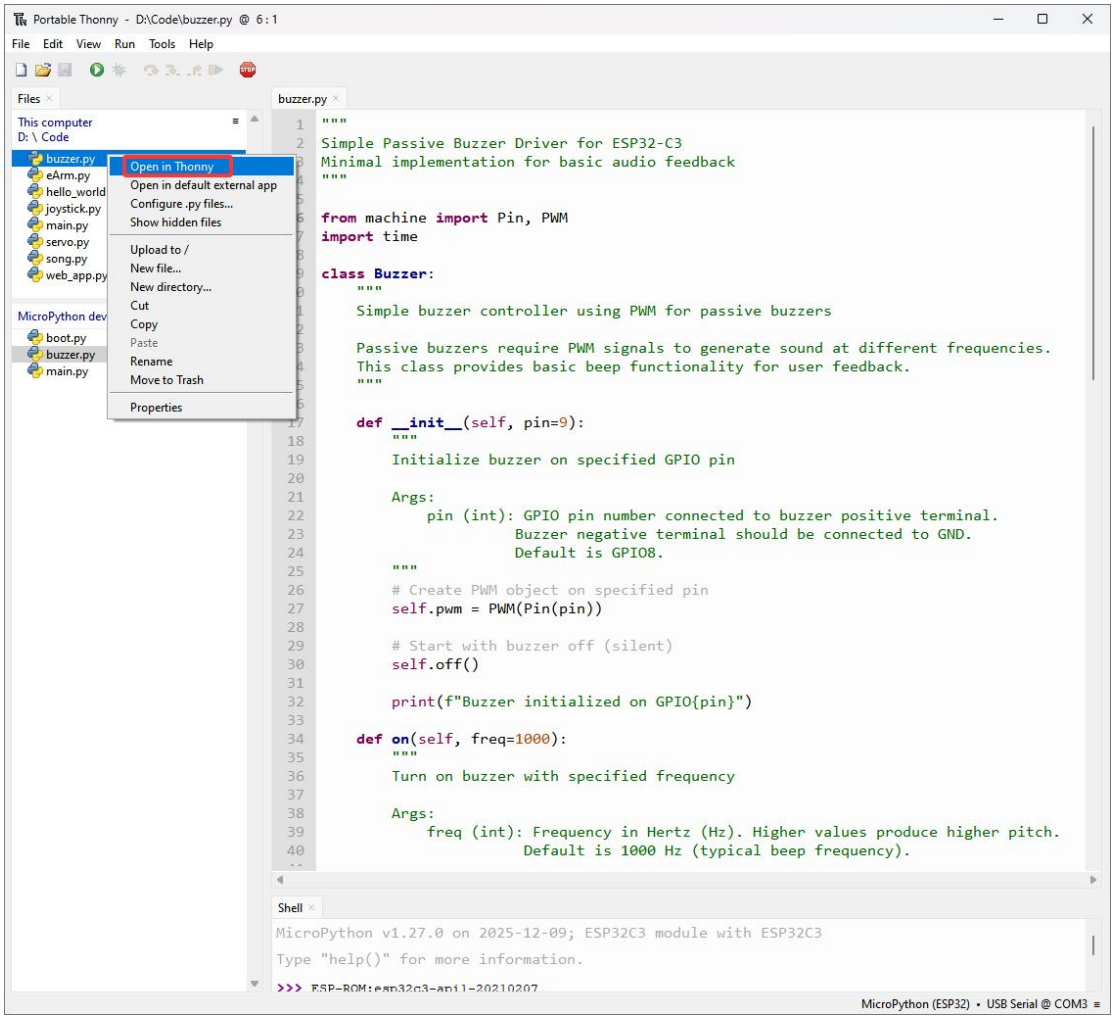


After the code is uploaded successfully, the buzzer on the ESP32-C3 board will keep making sounds:

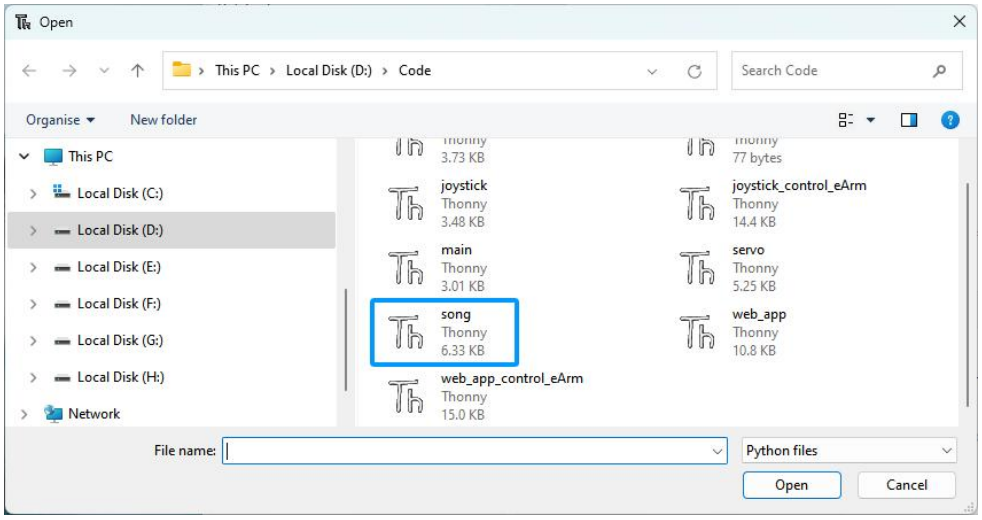


Code Explanation:

The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.

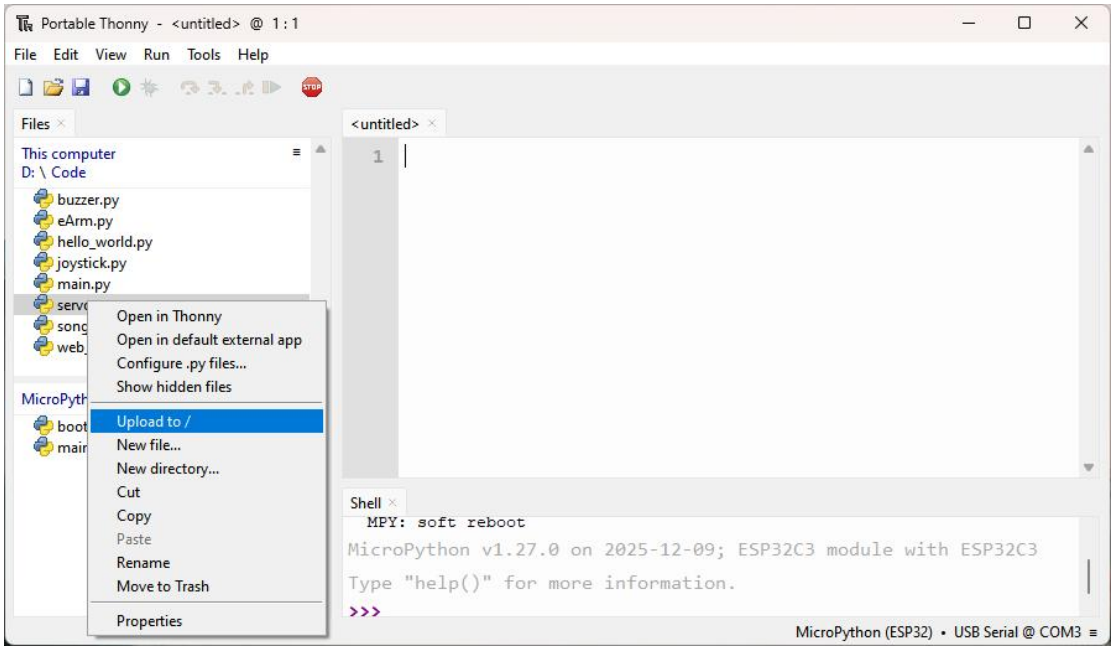


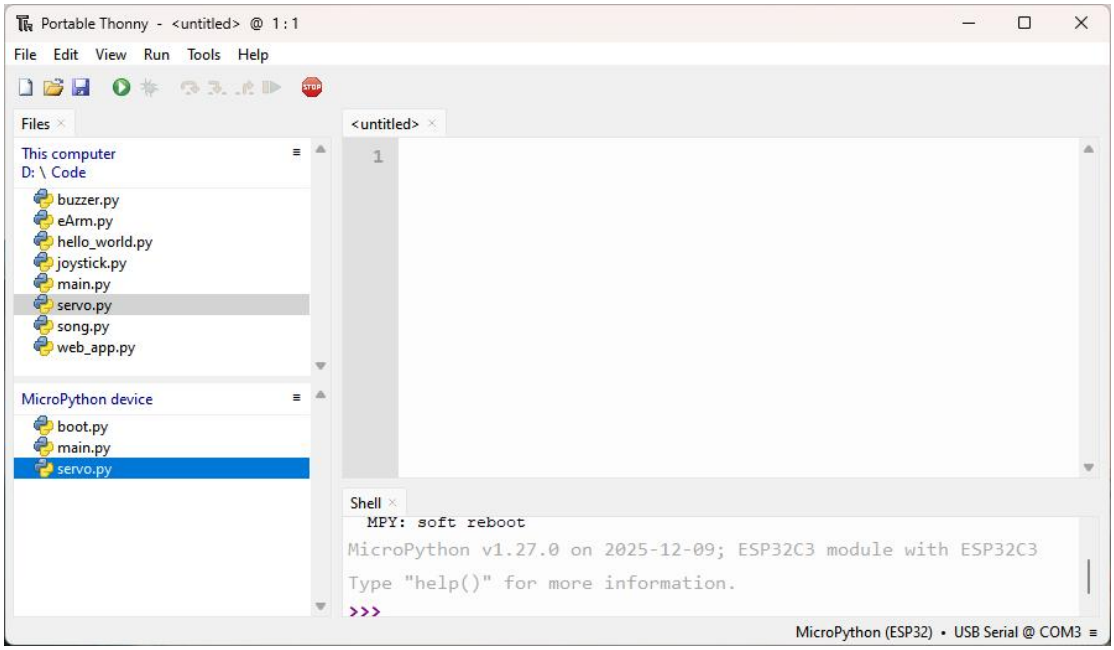
We also provide a sample code for a passive buzzer playing songs. If you are interested, you can follow the steps above to implement it.



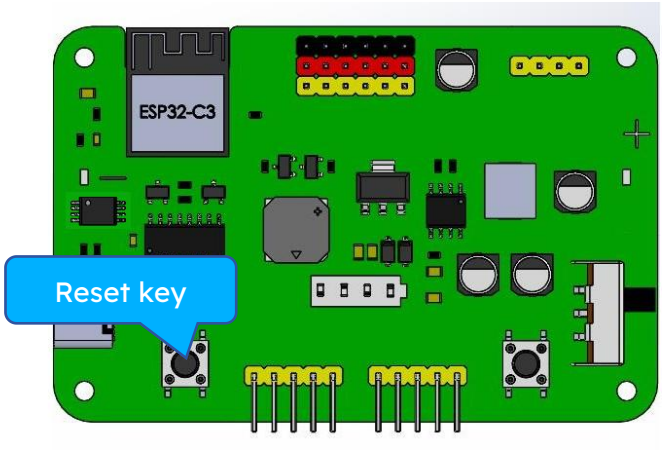
4.5.2 To Drive a Servo

Select “servo.py” , right-click your mouse and select “Upload to /” to upload code to ESP32-C3’s root directory.

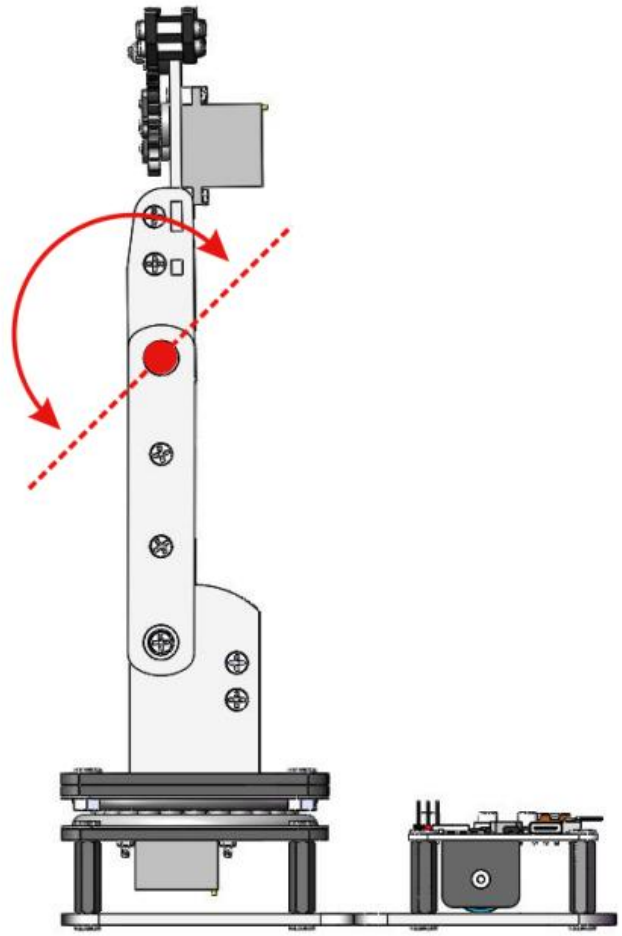




Press the reset key and in the box of the illustration below, you can see the code is executed.

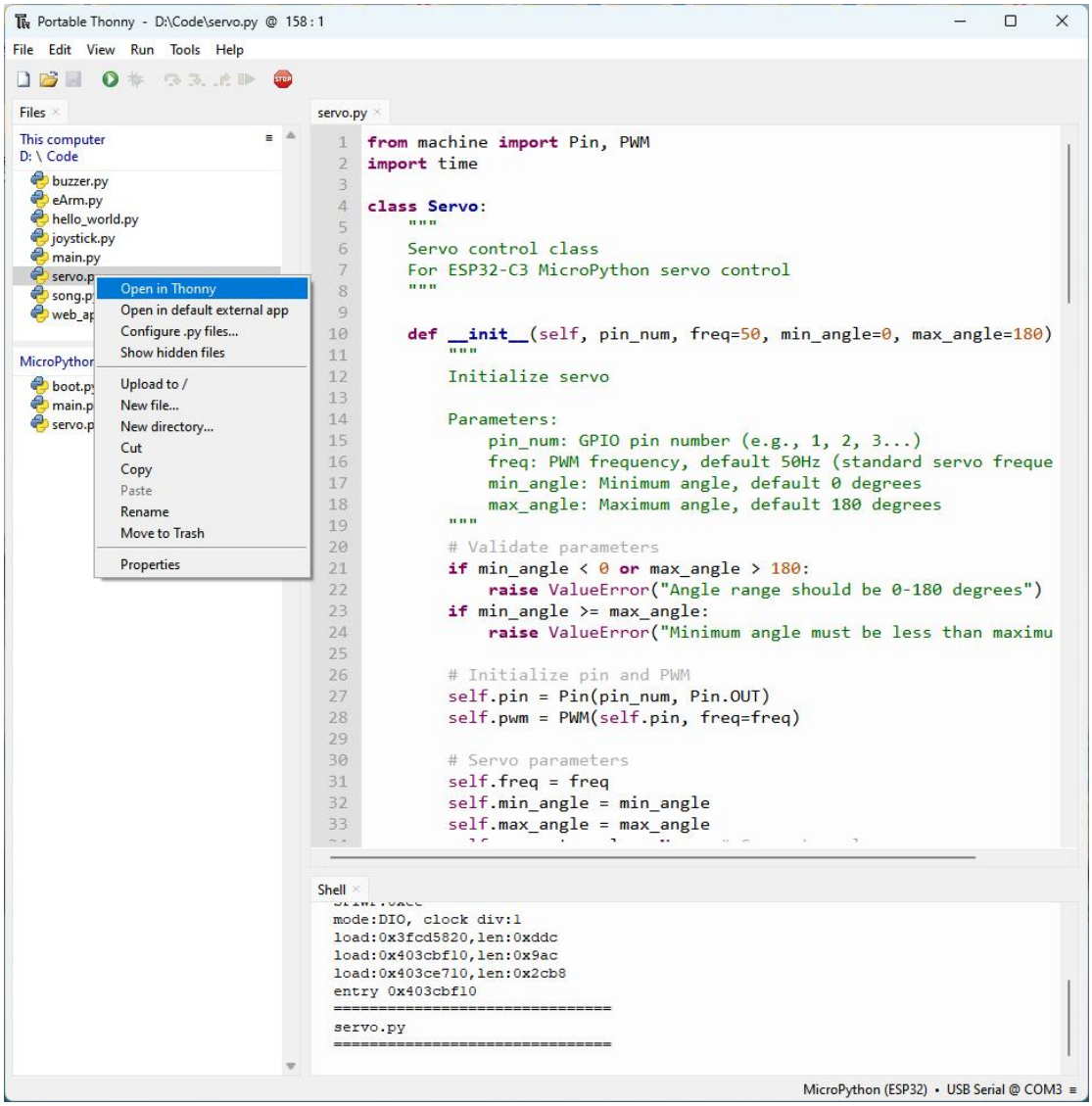


After the code is uploaded successfully, the eArm's servo C will swing 0-180, 180-0 in a cycle:



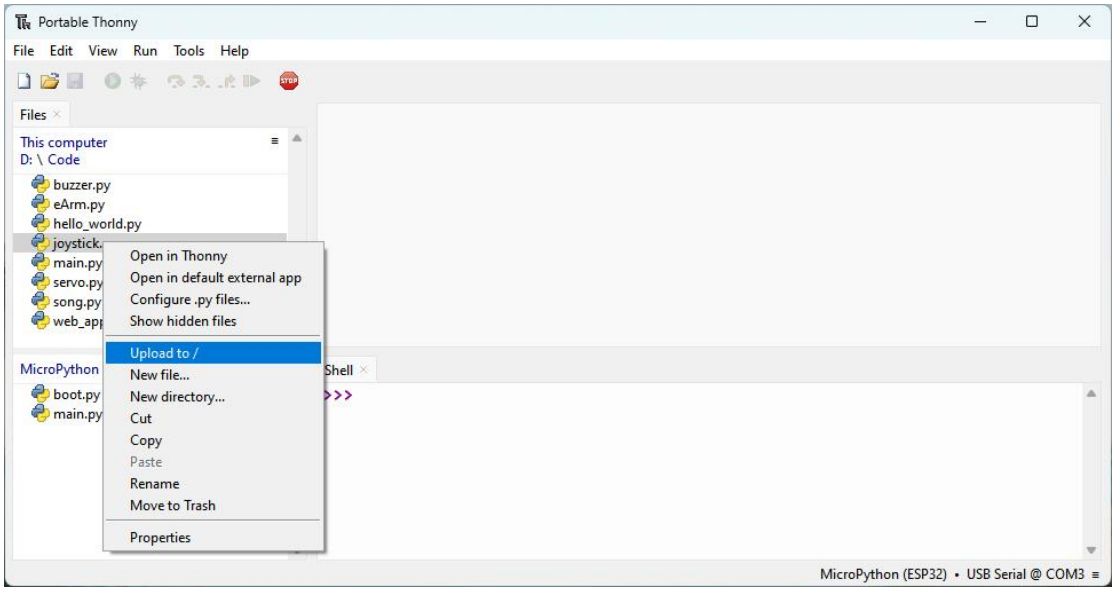
Code Explanation：

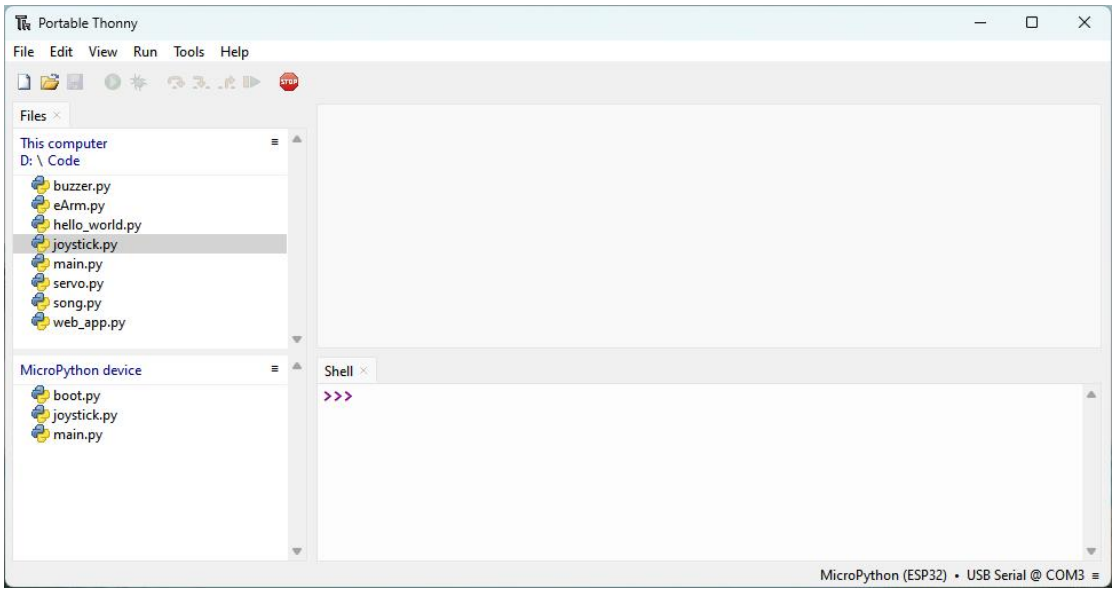
The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.



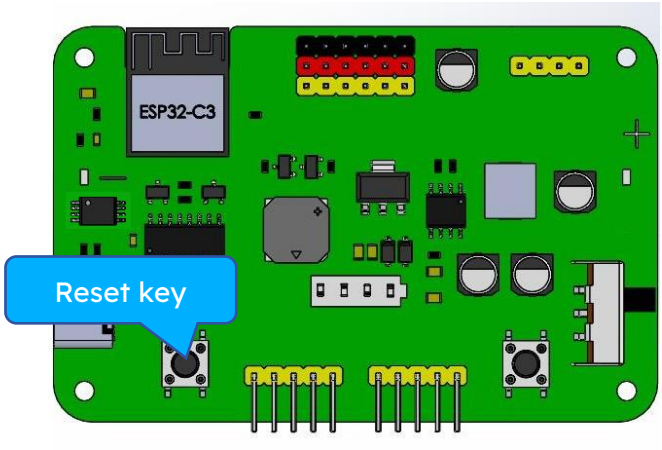
4.5.3 Read the Value of the Joystick

Select “joystick.py” , right-click your mouse and select “Upload to /” to upload code to ESP32-C3’s root directory.

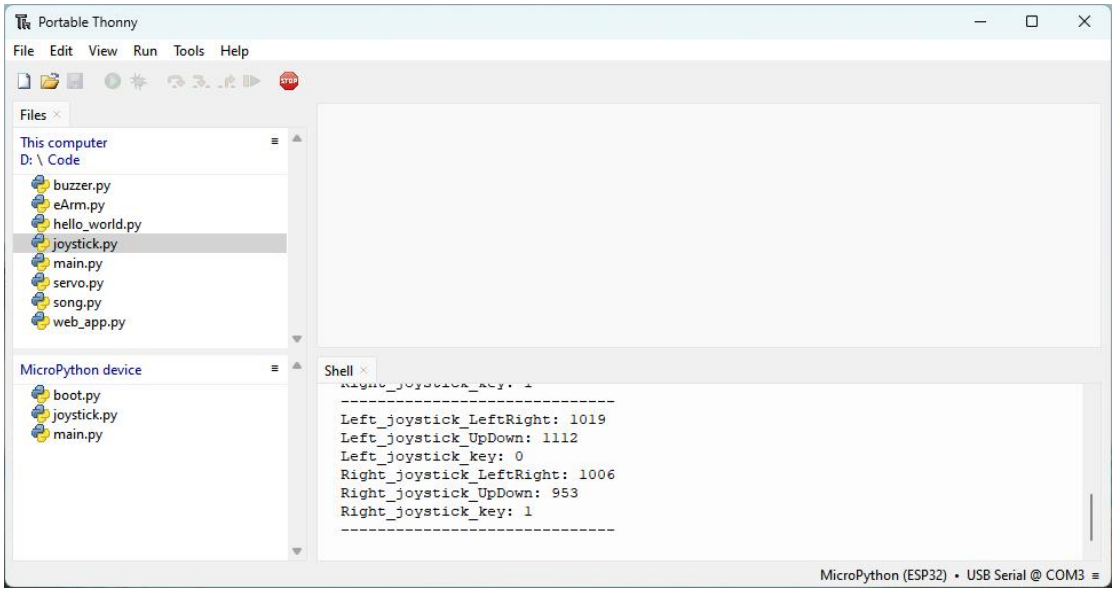




Press the reset key and in the box of the illustration below, you can see the code is executed.

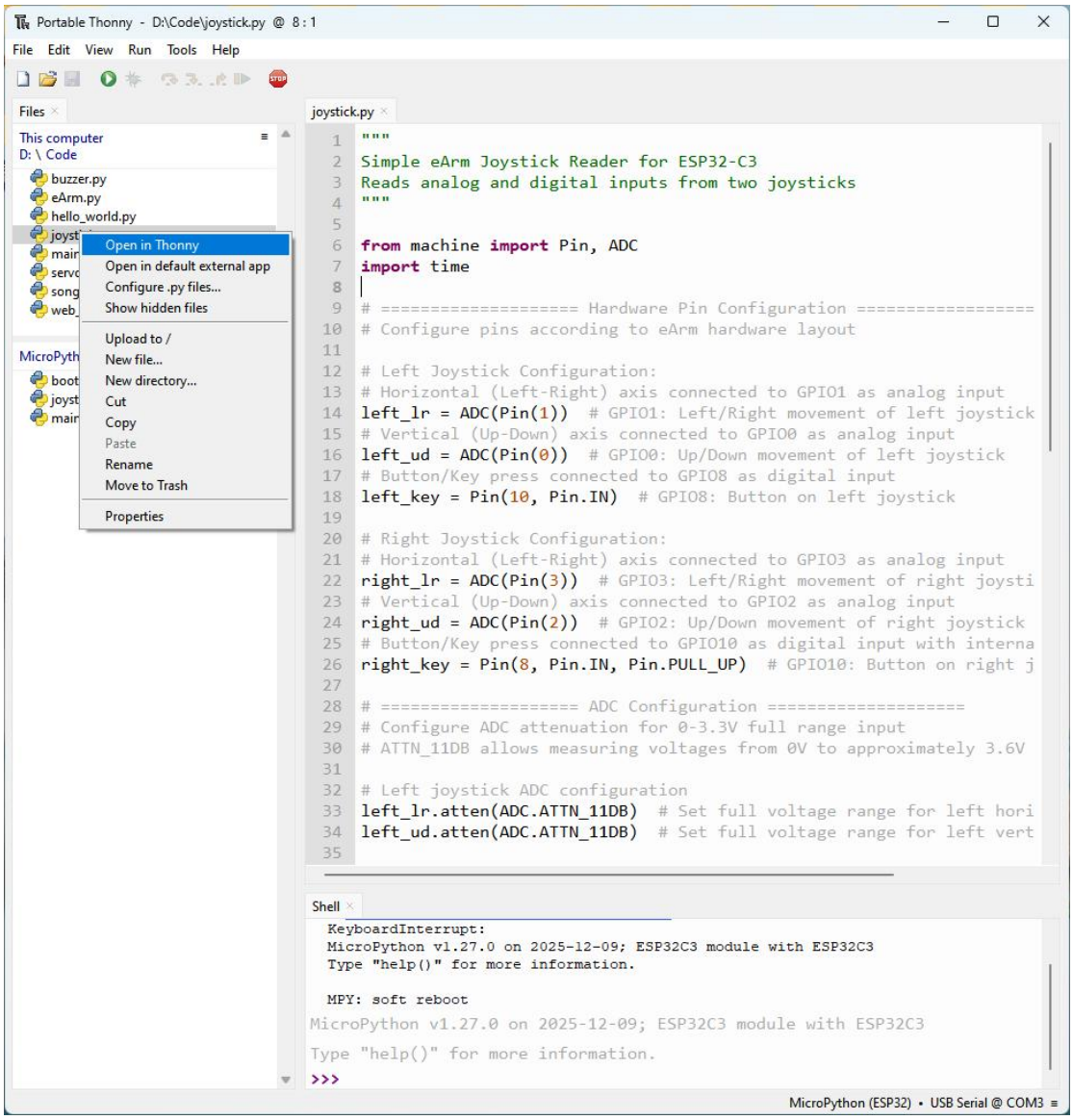


Shake the handle left, right, up, down, or press the joystick down, and the shell will print the joystick value:



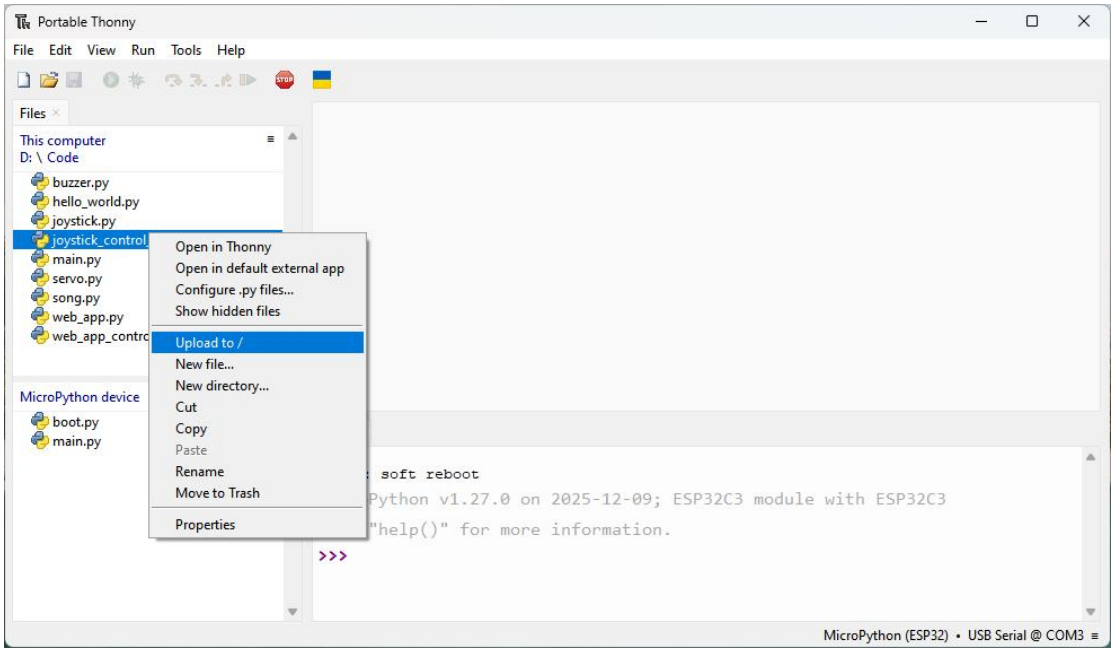
Code Explanation：

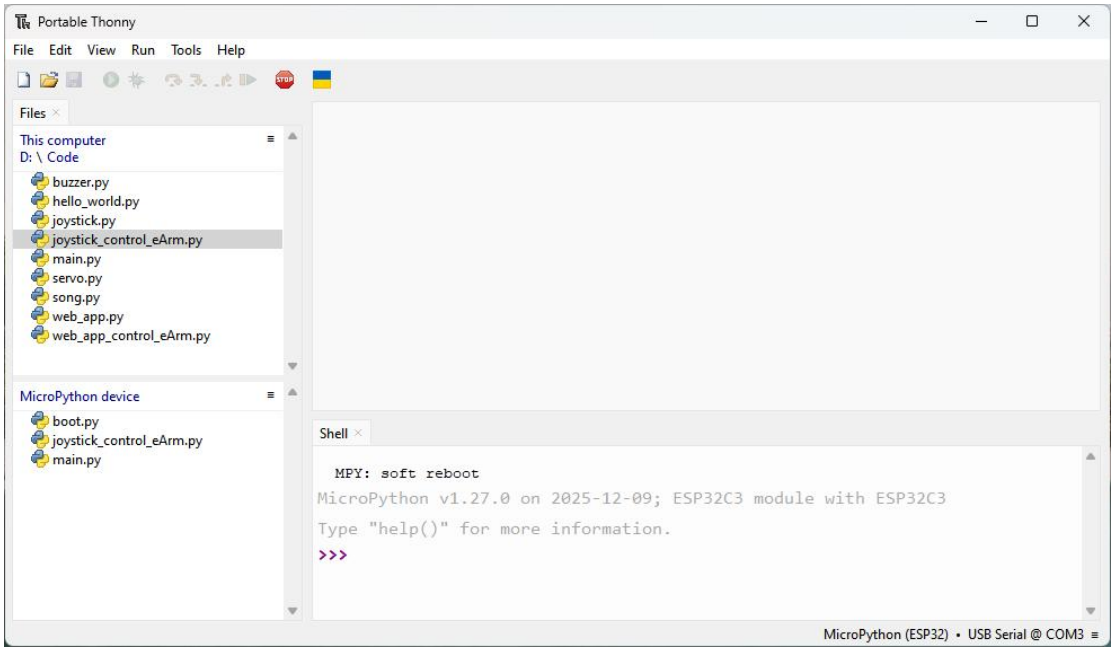
The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.



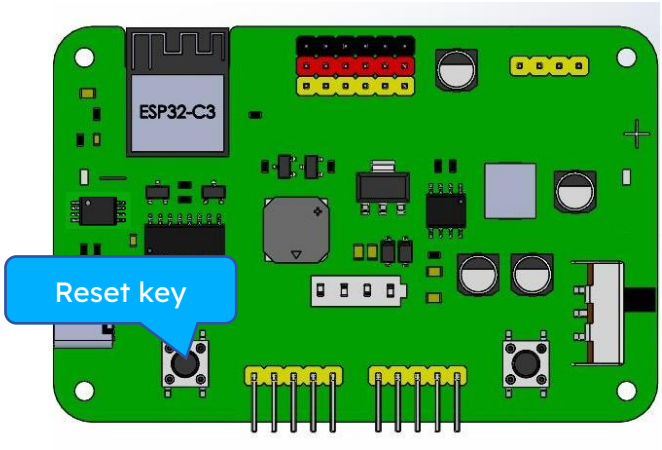
4.5.4 Joystick Control eArm

Select “joystick_control_eArm.py” , right-click your mouse and select “Upload to /” to upload code to ESP32-C3’s root directory.

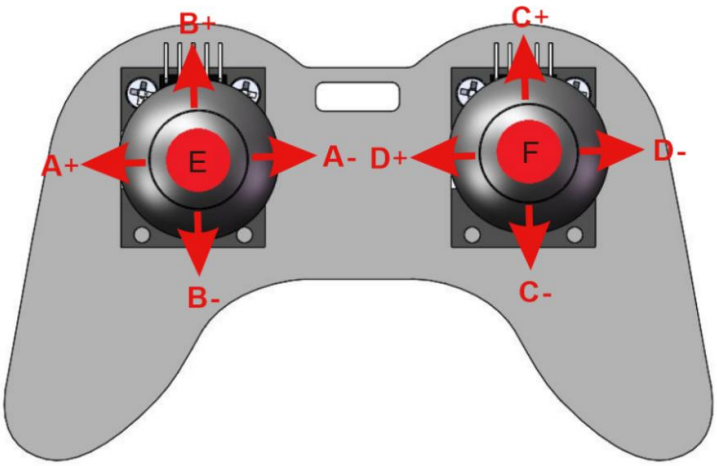




Press the reset key and in the box of the illustration below, you can see the code is executed.



Shake the handle left, right, up, down, or press the joystick down:

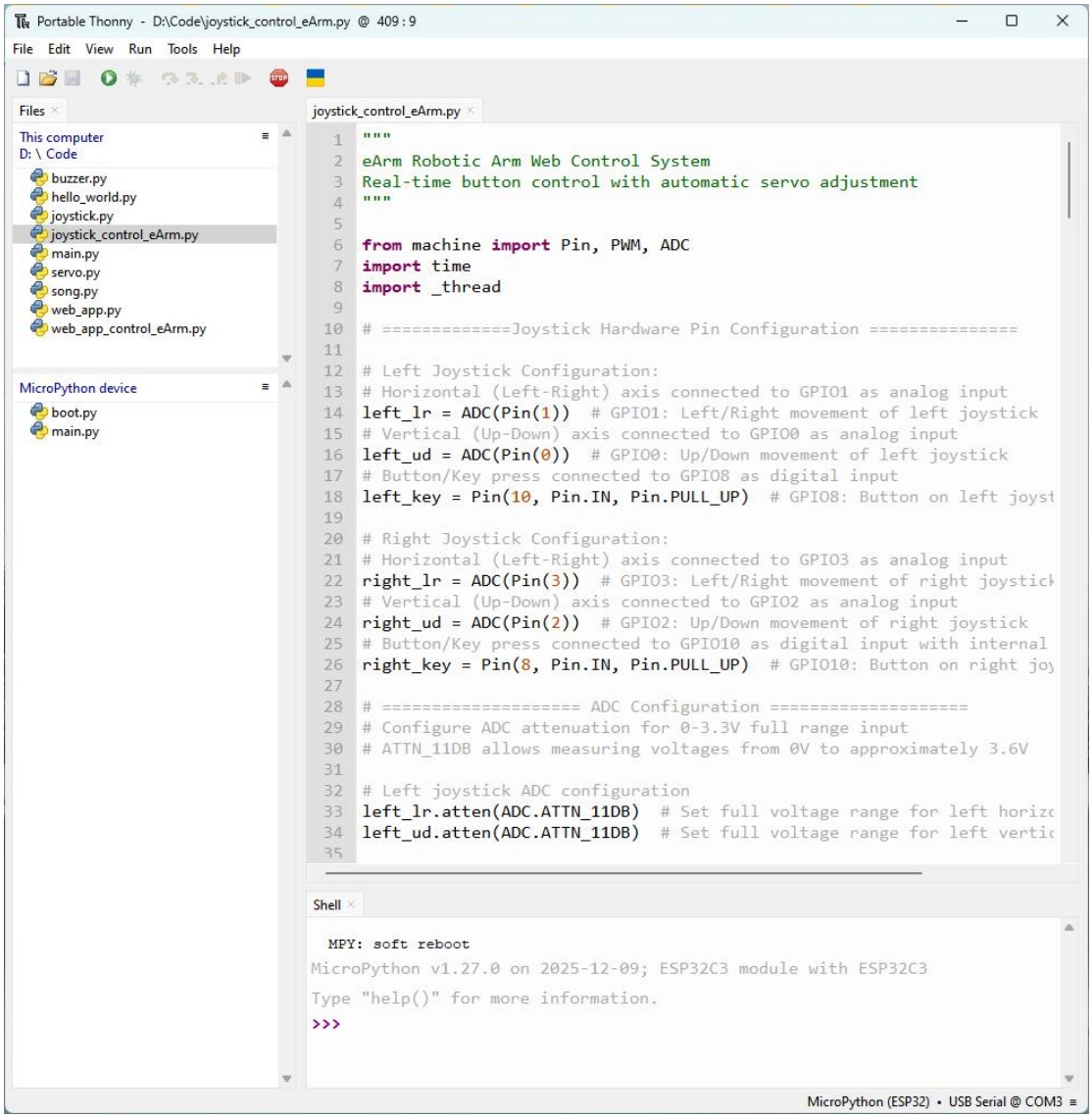


A+: Rotate arm left (Servo A)
A-: Rotate arm right (Servo A)
B-: Raise rear arm (Servo B)
B+: Lower rear arm (Servo B)
C-: Raise the forearm (Servo C)
C+: Lower the forearm (Servo C)
D+: Open the gripper (Servo D)
D-: Close the gripper (Servo D)

F: Reserve
E: Enable/disable the buzzer

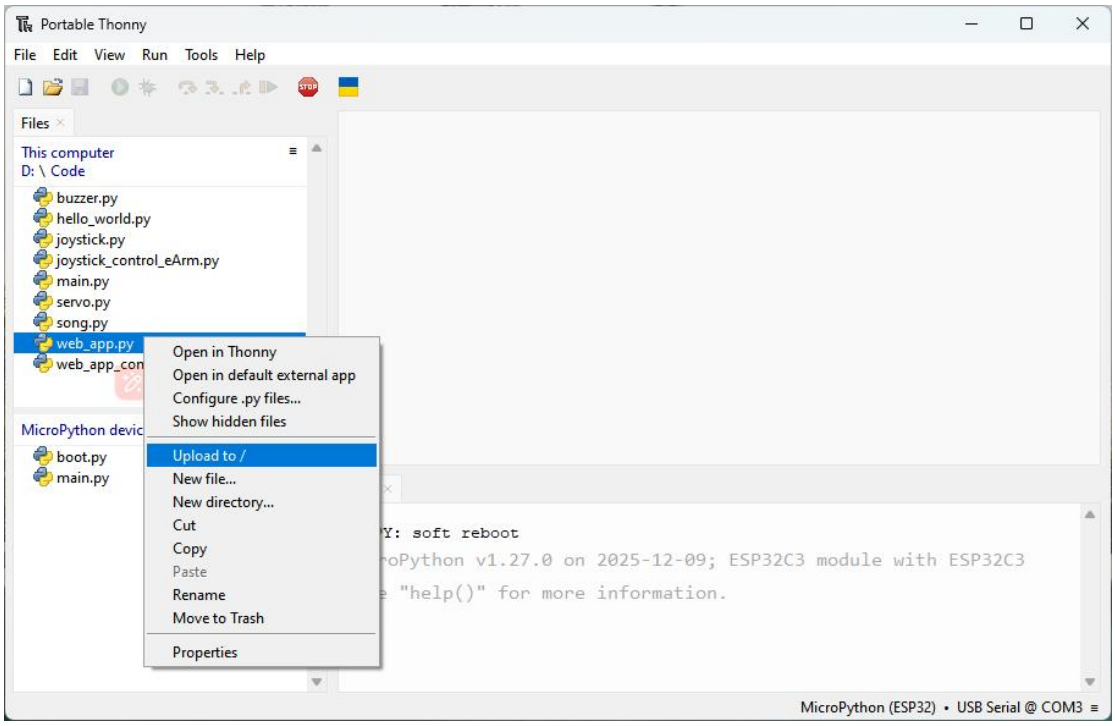
Code Explanation：

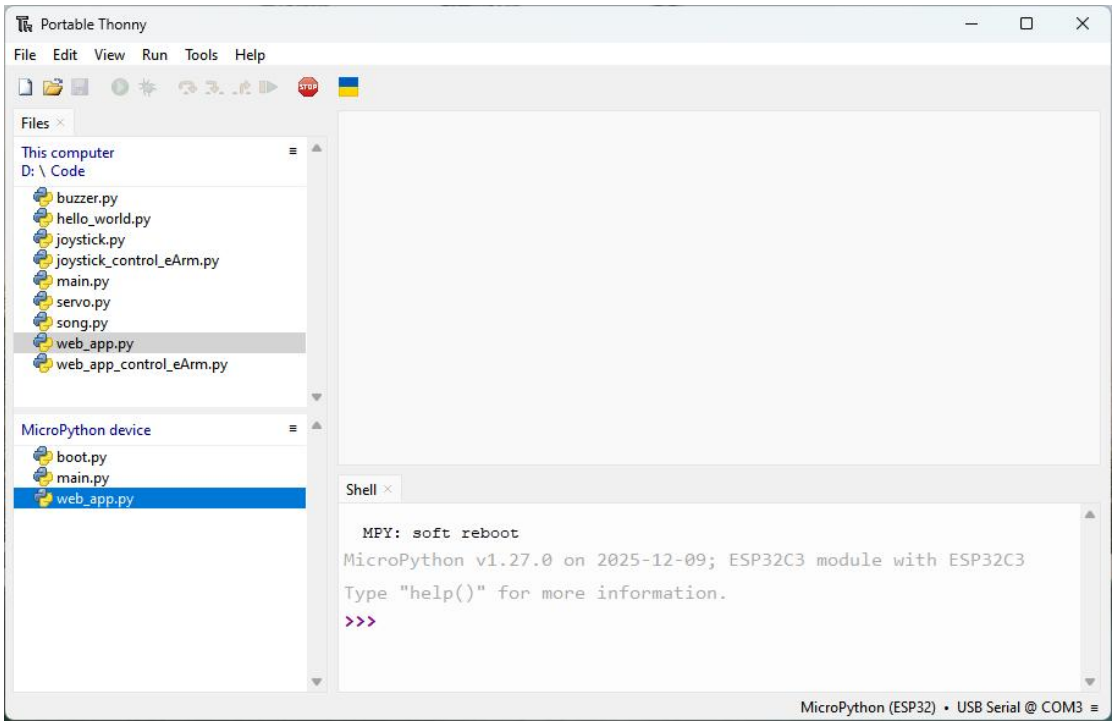
The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.



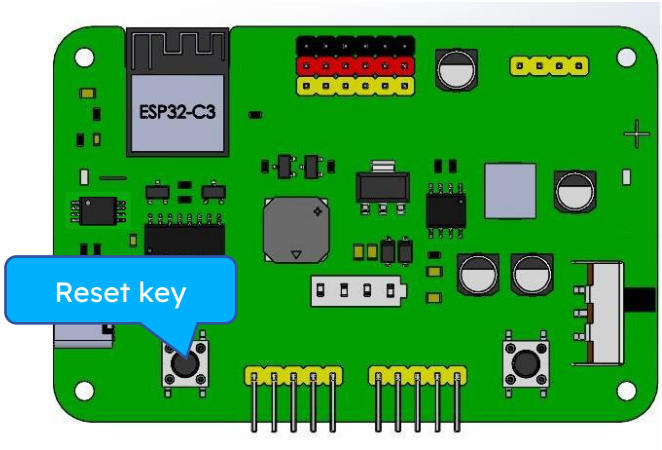
4.5.5 Read the Value of the Web APP

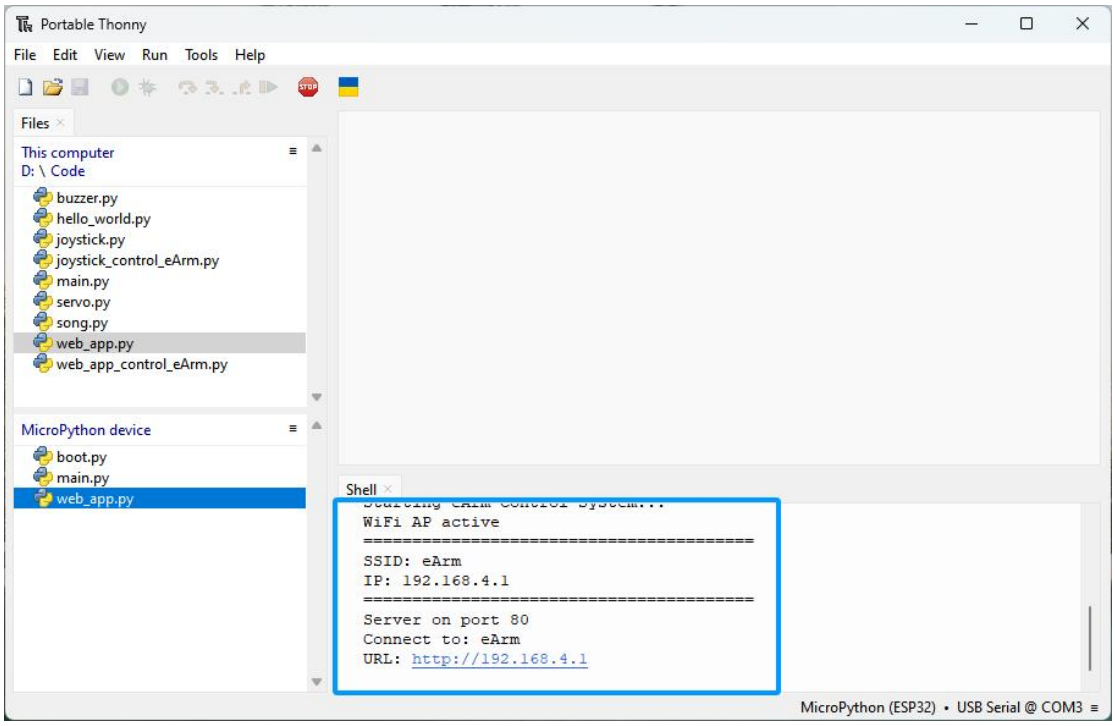
Select “web_app.py” , right-click your mouse and select “Upload to /” to upload code to ESP32-C3’s root directory.





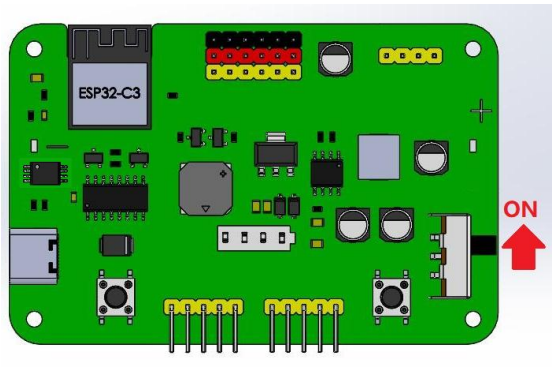
Press the reset key and in the box of the illustration below, you can see the code is executed.

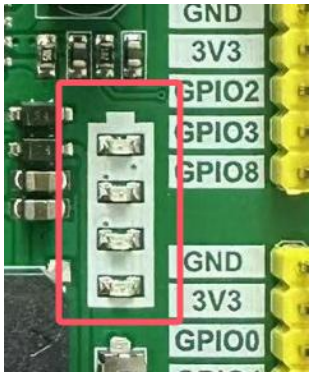




Turn the eArm's power switch to the ON position.

■ 1: Make sure the 18650 lithium battery is installed.
■ 2: The battery indicator shows 3 bars or more.

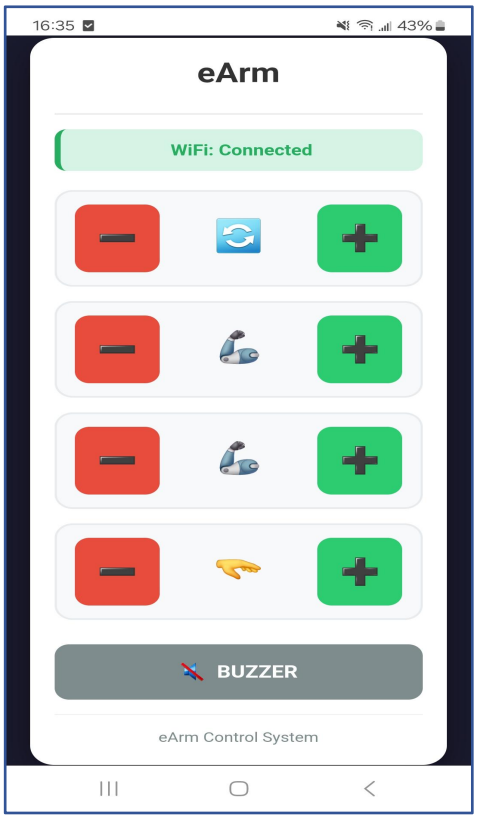




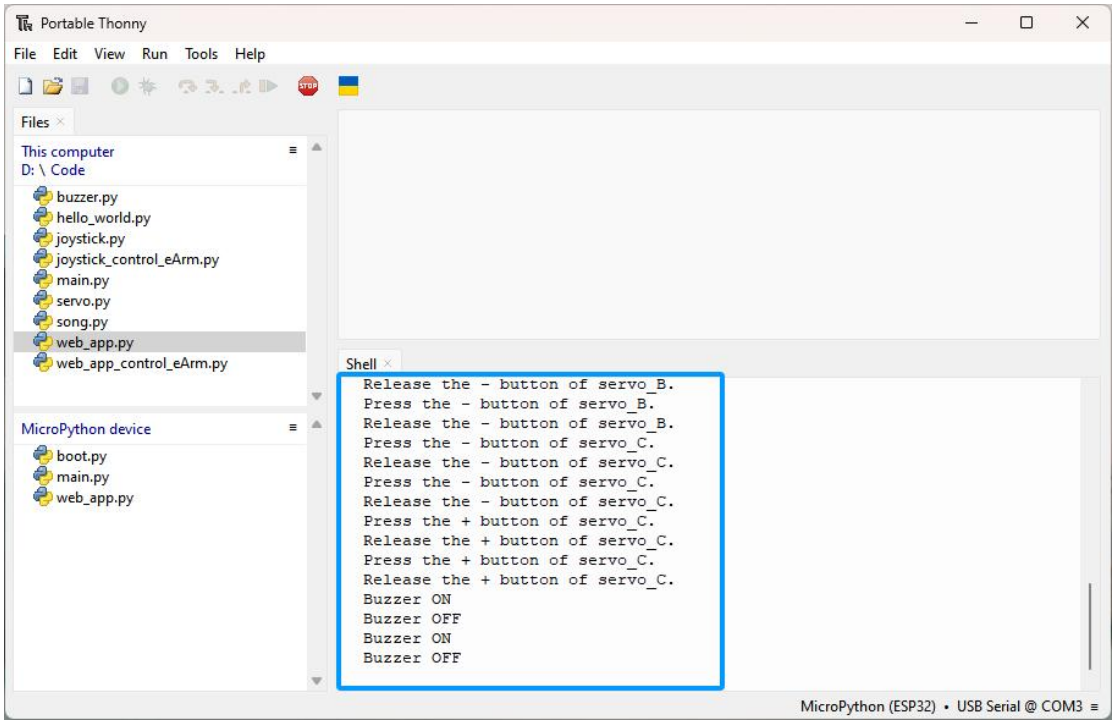
Turn on the phone's Wi-Fi and connect to the network named 'eArm'
■ : Once connected to the Wi-Fi, you may see a 'Network connection failed' message; just disregard it.
■ : Step 1: Turn off mobile data in your phone settings
■ : Step 2: This ensures stable connection to the Web App

1. Open the web browser on your phone
2. Type 192.168.4.1 into the address bar
3. Press Enter to connect

After successful connection, the control interface will appear in your browser - you can now operate the eArm!

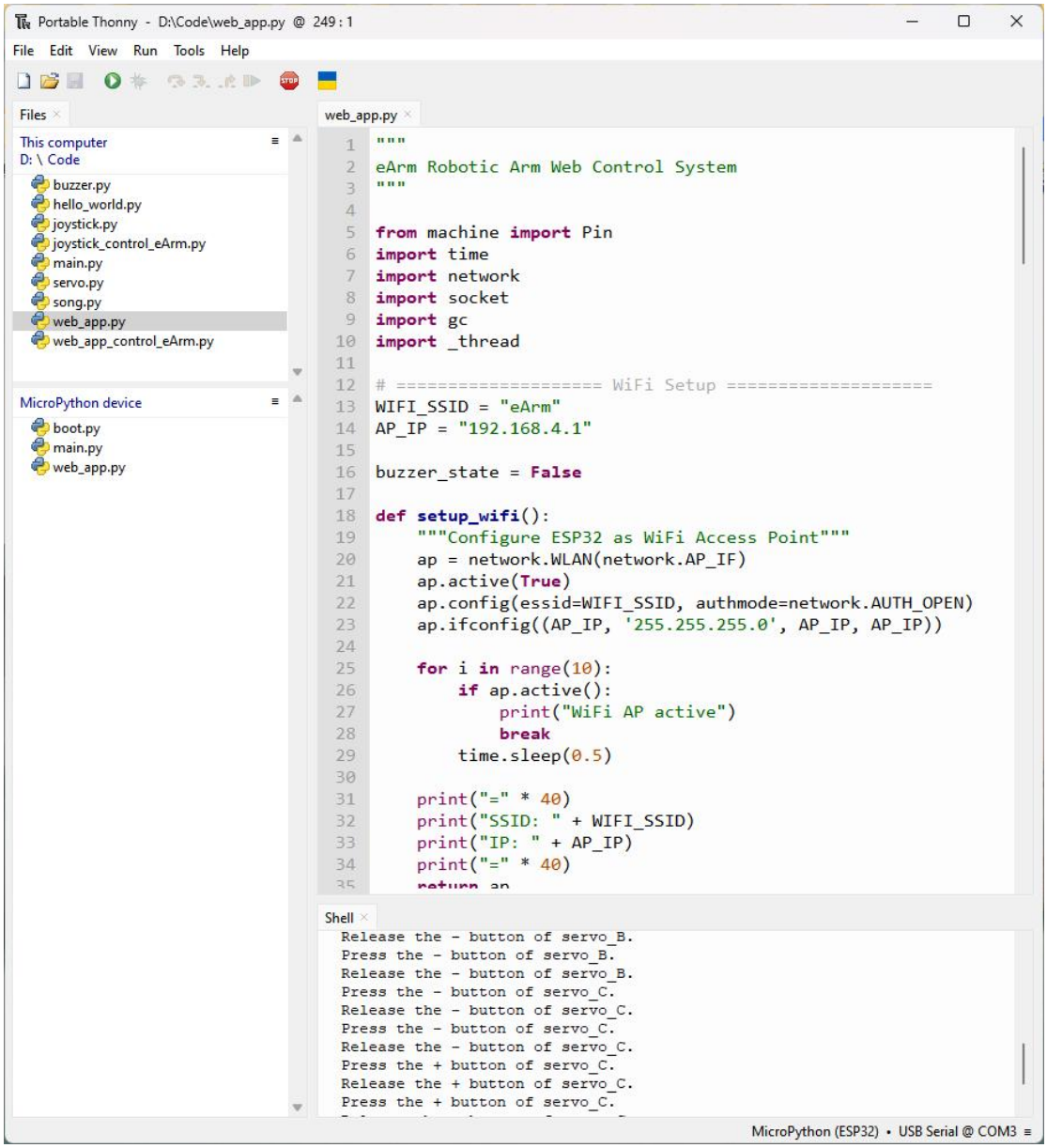


Click the button and the shell will print the key information.



Code Explanation：

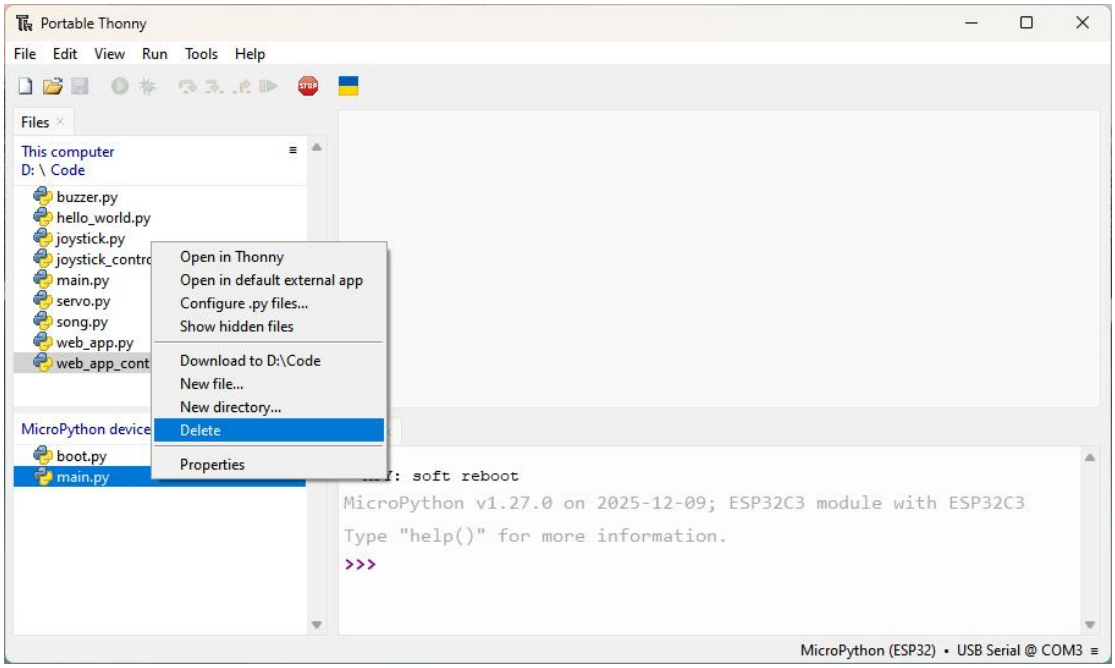
The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.



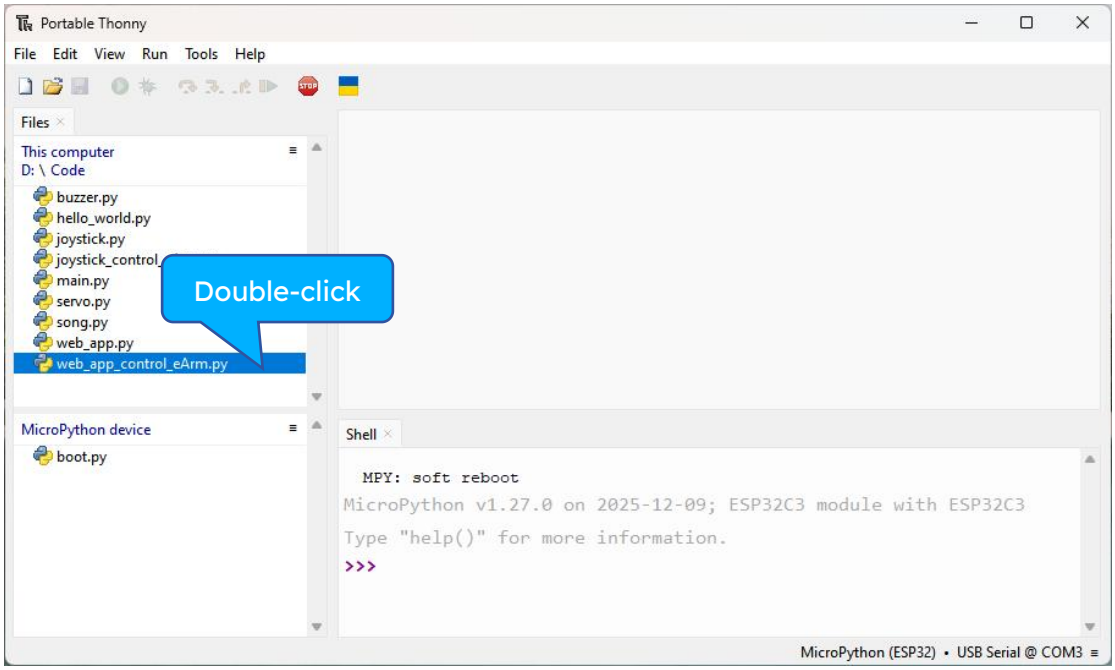
4.5.6 Web App Control eArm

By invoking other ".py" files from the script in "main.py", it is only suitable for simple code. For more powerful code, you need to delete the "main.py" file that contains the script and directly save the file you want to run as "main.py".

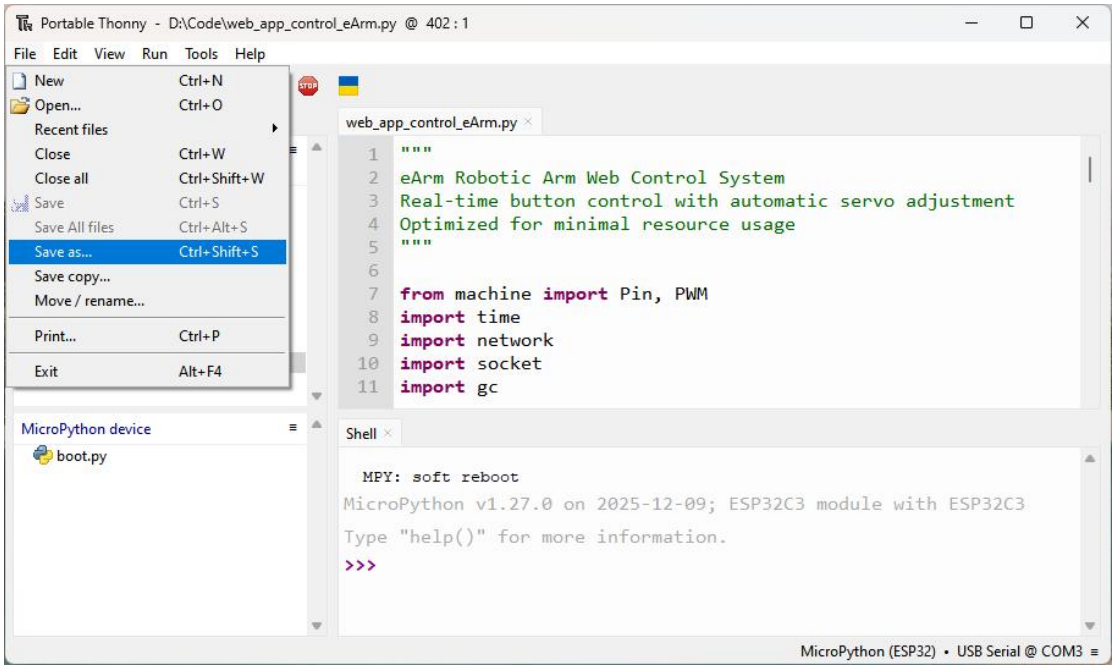
Delete the "main.py" file that contains the script:

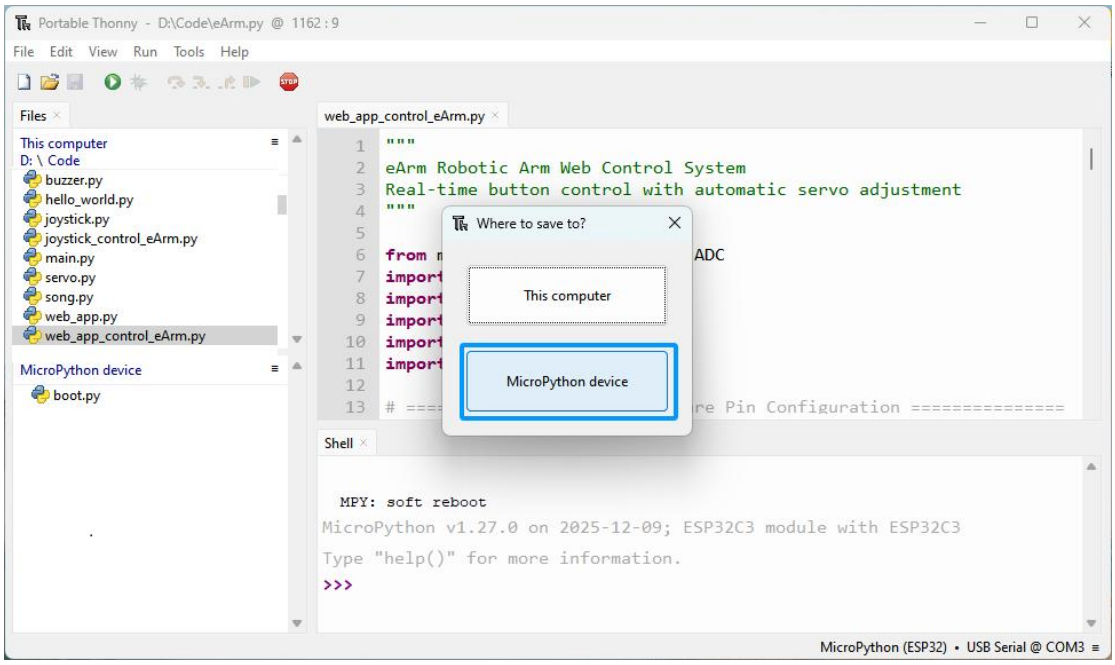


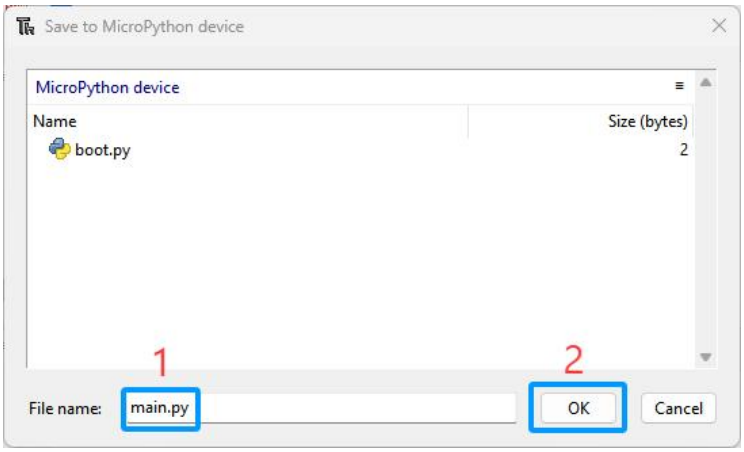
Double-click the "web_app_control_eArm.py" file to open it.

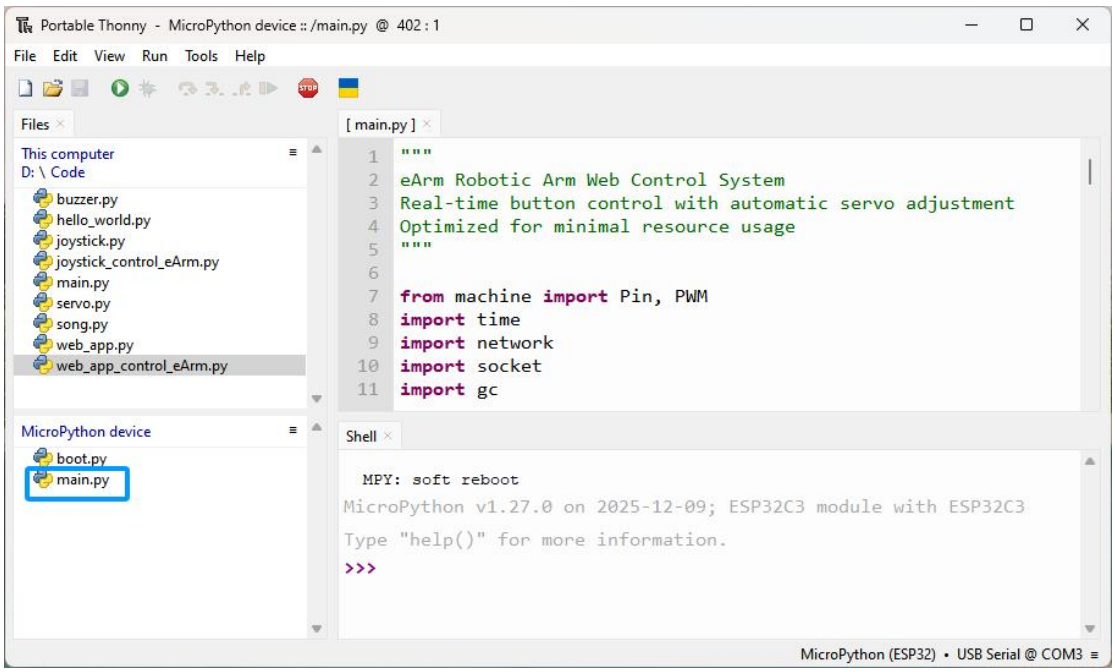


Click "File" > "Save as ...", then save "web_app_control_eArm.py" with the name "main.py" to the root directory of the ESP32-C3 device.

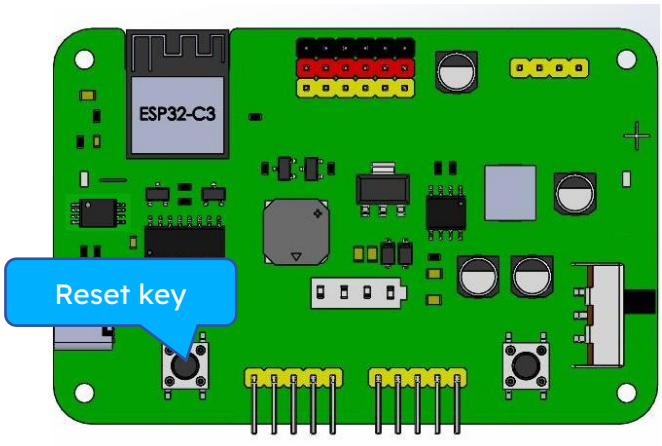






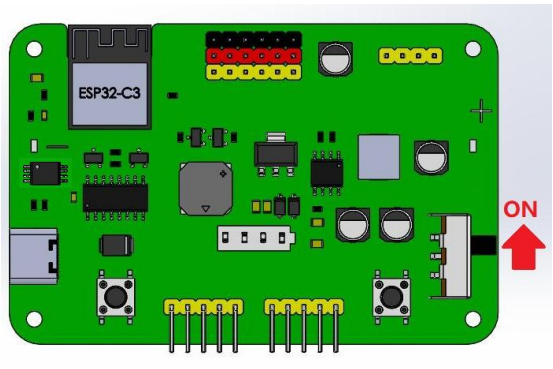


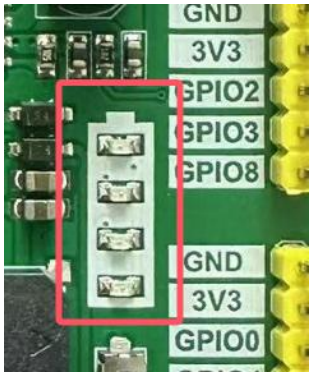
Press the reset key and in the box of the illustration below, you can see the code is executed.



Turn the eArm's power switch to the ON position.

■ 1: Make sure the 18650 lithium battery is installed.
■ 2: The battery indicator shows 3 bars or more.

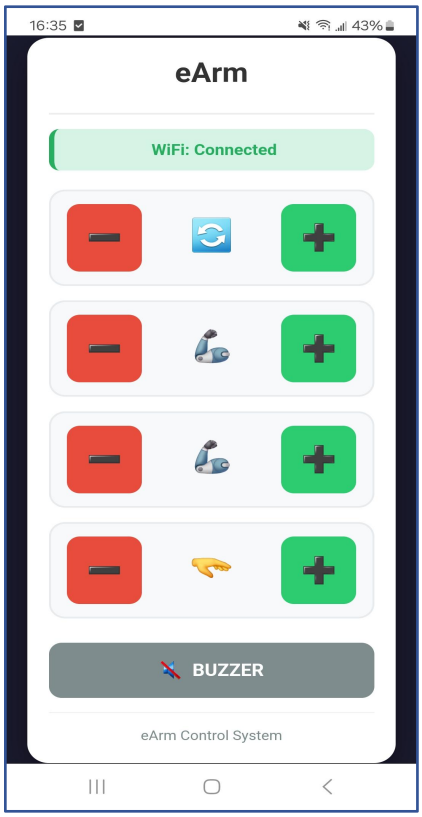


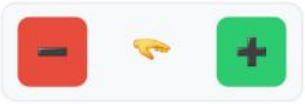


Turn on the phone's Wi-Fi and connect to the network named 'eArm'
■ : Once connected to the Wi-Fi, you may see a 'Network connection failed' message; just disregard it.
■ : Step 1: Turn off mobile data in your phone settings
■ : Step 2: This ensures stable connection to the Web App

1. Open the web browser on your phone
2. Type 192.168.4.1 into the address bar
3. Press Enter to connect

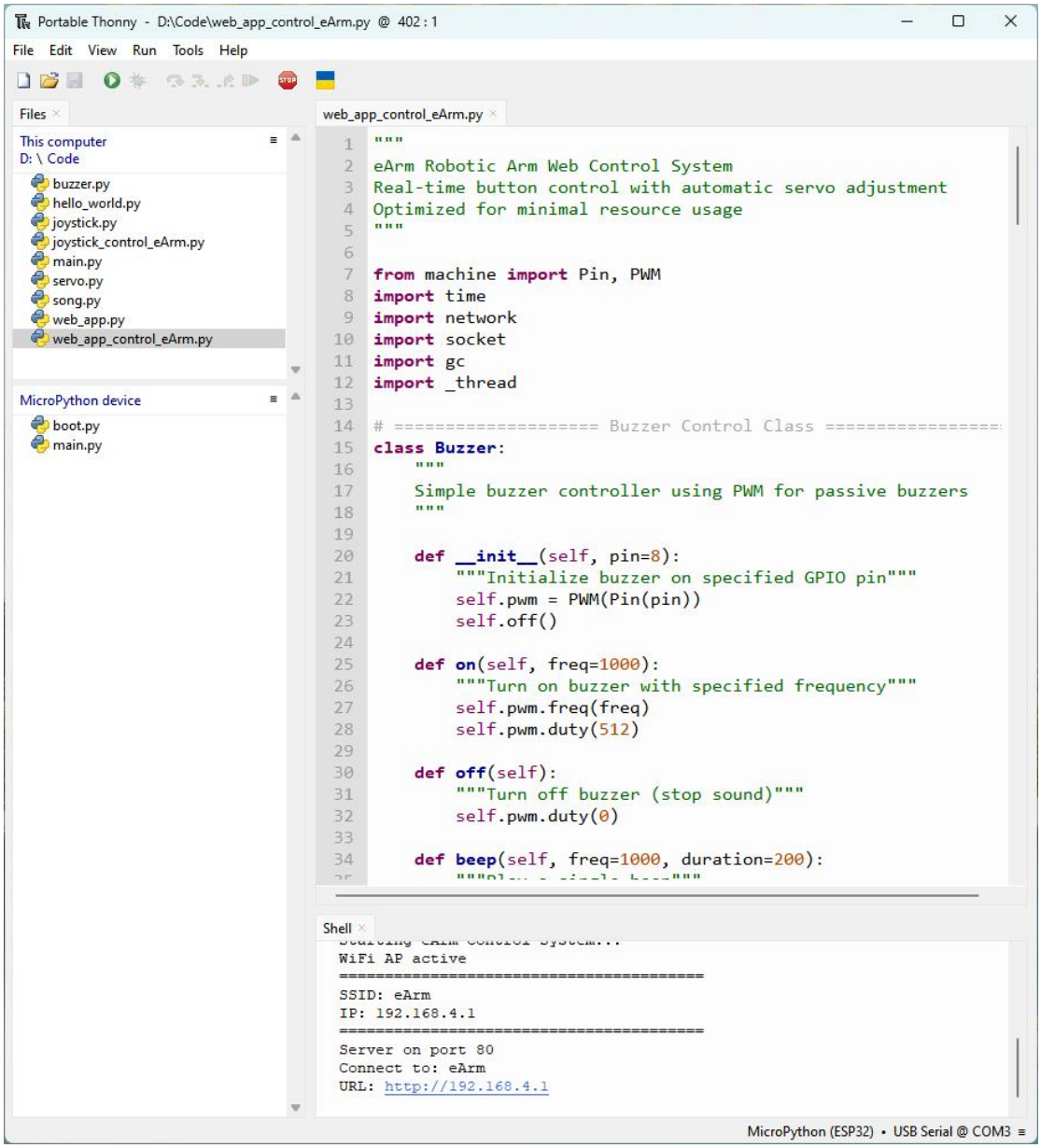
After successful connection, the control interface will appear in your browser - you can now operate the eArm!



		Robot arm turns left
		Robot arm turns right
		Rear arm raises
		Rear arm lowers
		Front arm raises
		Front arm lowers
		Gripper opens
		Gripper closes
		Buzzer ON
		Buzzer OFF

Code Explanation：

The detailed analysis of the code is in the ".py "file. Please open and read it with Thonny IDE.



5

Factory Reset Function

5.1 Firmware

Providing the device's "factory reset program" directly to users in the form of a firmware file (e.g., .bin or .hex), allowing them to flash it into the device without setting up an Arduino development environment, thereby completing the restoration process.

1. Issues with the Traditional Approach

Typically, restoring a device via Arduino requires users to:

- Install the Arduino IDE or related development tools.
- Download the source code (.ino file), possibly requiring additional library setups and dependencies.
- Compile and upload it to the device.

This process has a high barrier for non-technical users and may fail due to environment-related issues.

2. Advantages of Providing Firmware Directly

Simplified User Process:

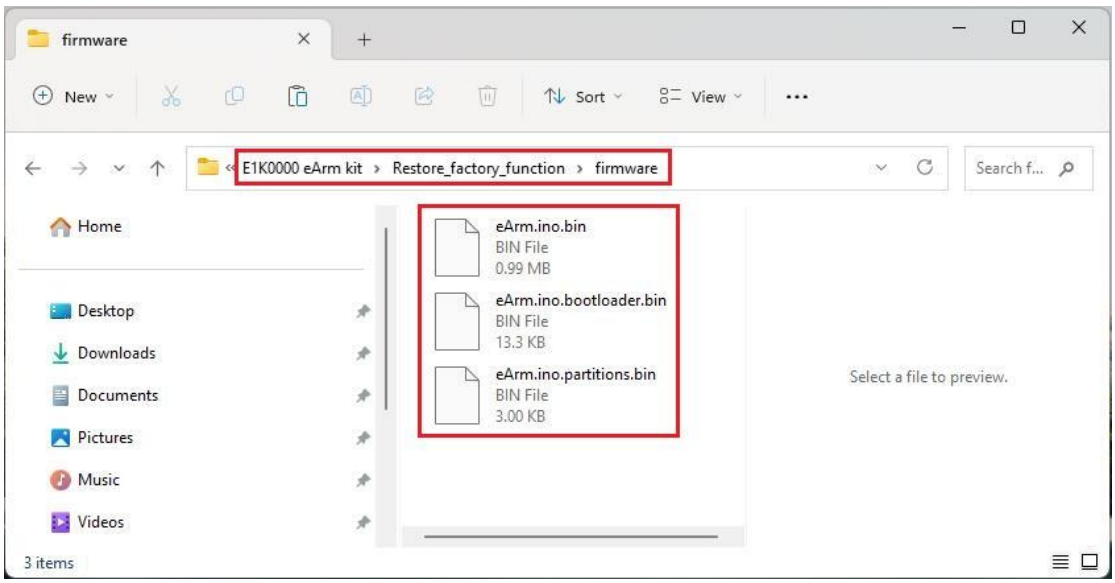
- Provide a pre-compiled firmware file (e.g., firmware.bin), allowing users to flash it with a simple tool (e.g., a serial flasher) in one step.
- No need to install Arduino IDE, resolve compilation errors, or manage library conflicts.

Use Cases:

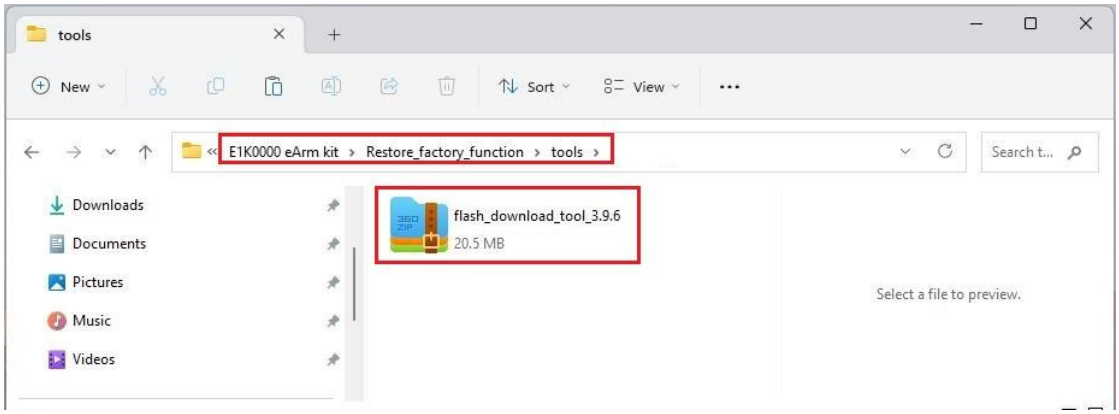
- Restoring a malfunctioning device to factory settings.
- Mass-flashing the same firmware to multiple devices (improves efficiency).

3. How to Implement This?

We provide factory firmware files for this product, which are saved in the following folders:



5.2 Burning Tool



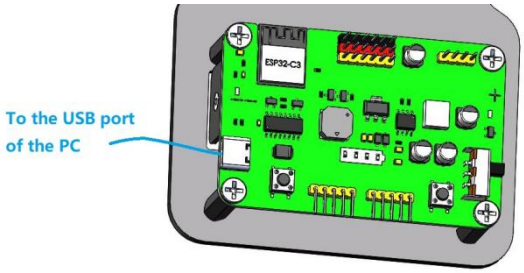
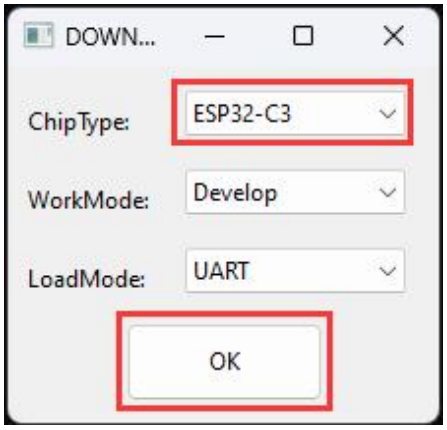
You can also download the latest version of the burning tool from the official website:
https://www.espressif.com.cn/zh-hans/support/download/other-tools

Flash Download Tools

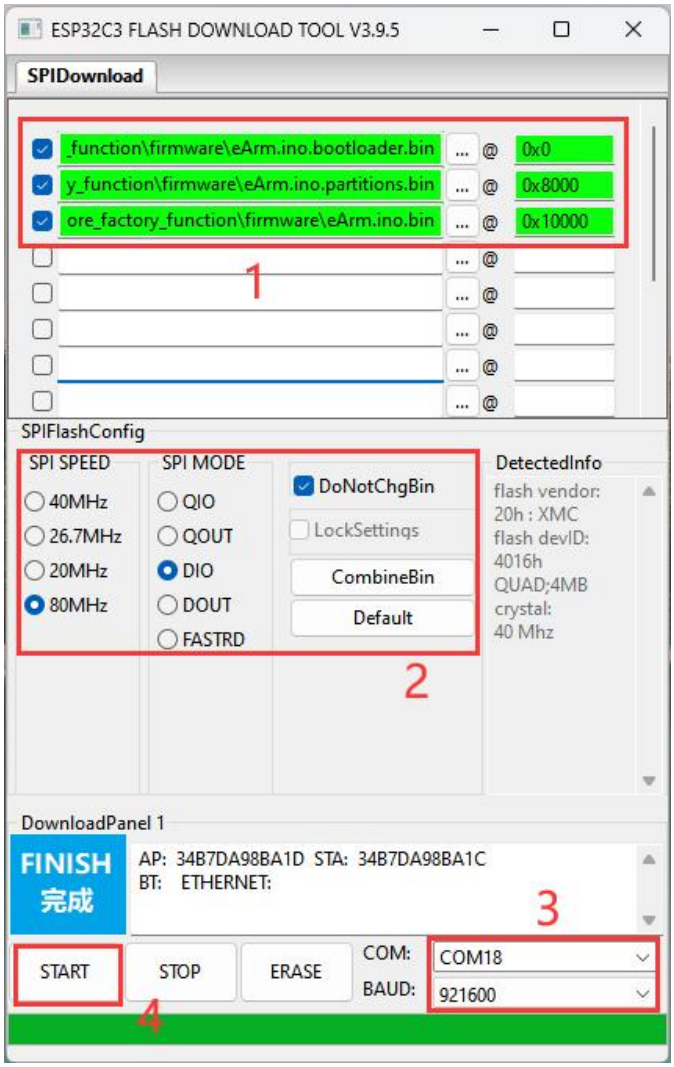
Expand all + Download selected

	Title	Platform	Version	Release Date	Download
+	Flash Download Tools	Windows PC	V3.9.7	2024.06.07	

5.3 How to Burn Firmware

Use a USB cable to connect the PC and eArm, as shown below:	Start the burning tool:
	

Select the firmware we provided and fill in the burning address correctly:



You must have installed the CH340 driver and turn on the switch of the ESP32 board, otherwise the COM port will not be found!

6

QA

6.1 Unable to recognize USB serial ports

☛Do you use a USB cable with data communication function?

☛Does the USB cable connect well?

☛Is the CH340 USB driver installed?

6.2 Robotic arm doesn't work

☛Whether to turn on the power switch during installation to initialize the angle of the servo.

☛Is the battery electricity sufficient?

6.3 Other problems

☛Please check whether the assembly is correct?

☛Please check whether the battery power is sufficient?

☛Please check whether the battery used in accordance with the specifications?

7
Contact Us

If you encounter any technical issues or wish to share feedback with us, please feel free to contact us

✉ support@siyeenove.com

https://siyeenove.com