MicroPython development environment construction for ESP32

CONTENTS

1. Download the Thonny software installation package

Choose MicroPython for ESP32 development, you must first choose a development tool software, in the Windows system can choose the development tool software are: Thonny, VS Code, PyDev, Pycharm, etc. Thonny is an IDE for beginners, it is easy to get started, and the features are basically enough. Although other development tools software is powerful, but for beginners, it is difficult to get started, of course, if there is a certain development foundation of scholars, you can choose these powerful, difficult IDE.

The Thonny software package can be downloaded directly from the official website.

Official website address: https://thonny.org/

After entering the official website page, you can see the latest version number, and provide a variety of computer system versions for choice, as shown below:

Figure 1.1 Thonny software package download page 1

Select the appropriate version to download according to your computer system (moving the mouse over the corresponding computer system name will pop up the software version for selection). Windows system is used here, then move the mouse to "**Windows**", the version download interface will pop up, as shown in the following picture.

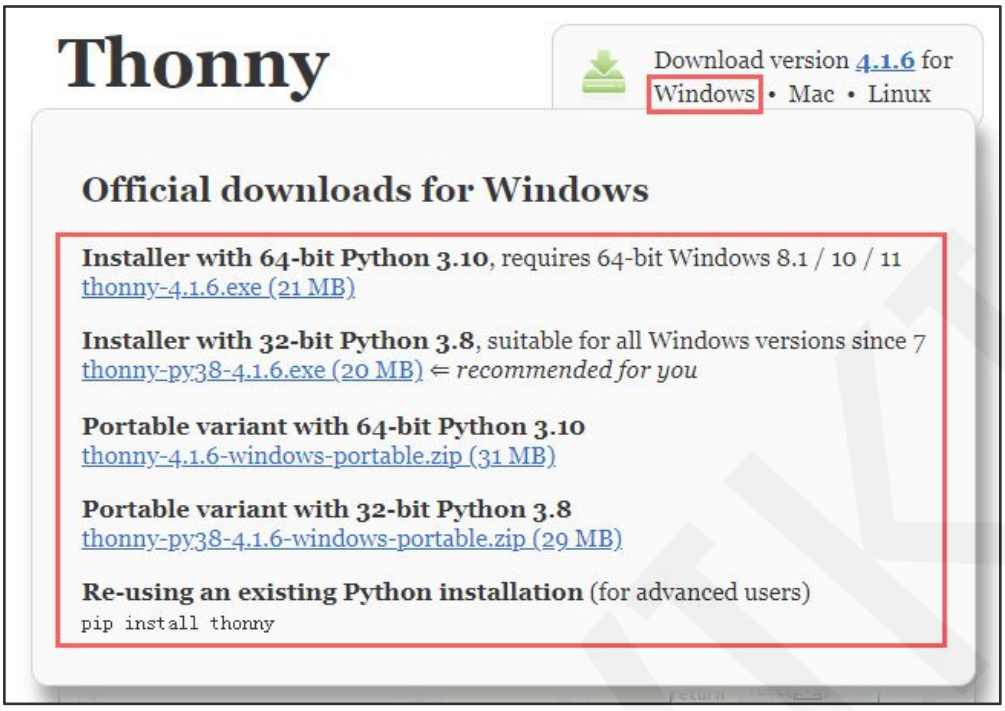

Figure 1.2 Thonny software package download page 2

The installation packages are described as follows:

The following two installation packages need to be installed step by step by using Installer, including installing the Python environment and Thonny software.

Installer with 64-bit Python 3.10 (For Windows 8 and above)

Installer with 32-bit Python 3.8 (For Windows 7 and later systems）

The following two installation packages already contain the Python environment and Thonny software. You do not need to install them. You can decompress them after downloading them.

Portable variant with 64-bit Python 3.10 (For Windows 8 and above)

Portable variant with 32-bit Python 3.8 (For Windows 7 and later systems）

To install only the Thonny software (the Python environment has been installed), run the "**pip install thonny**" command.

Re-using an existing Python installation

This section describes the **Installer** installation method only.Click "**Installer with 32-bit Python 3.8**" to download the Installer package, as shown in the following figure. Select the directory where the installation package is stored.

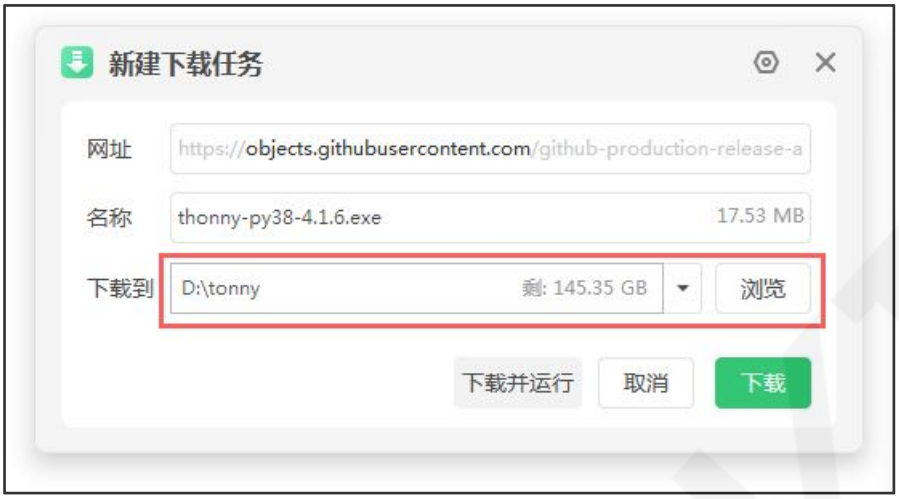

Figure 1.3 Thonny software installation package download task

2. Thonny software installation

After the software installation package is downloaded, open the save folder, and then double-click the exe file to enter the program installation (if the pop-up window asking whether to run the file, directly click the "**Run**" button), as shown in the following picture:

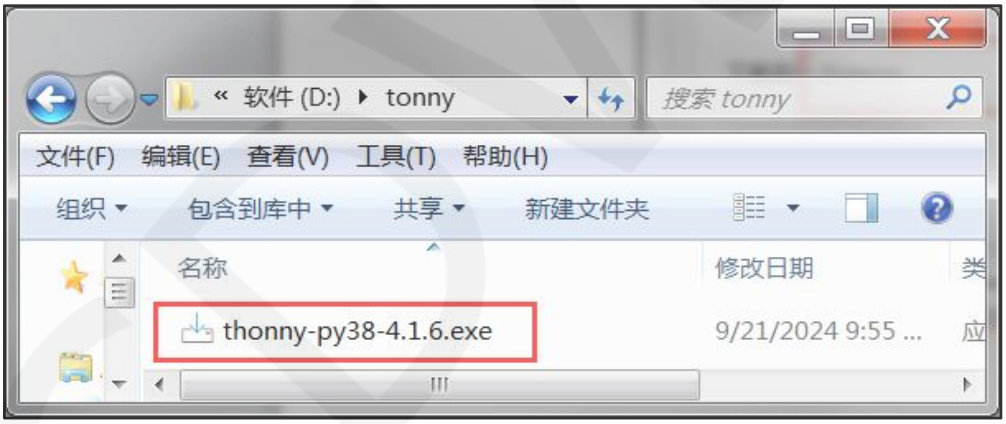

Figure 2.1 Thonny software installation package exe file

At the beginning, the installation mode selection screen pops up. Here, choose to install only for individual users, as shown below:

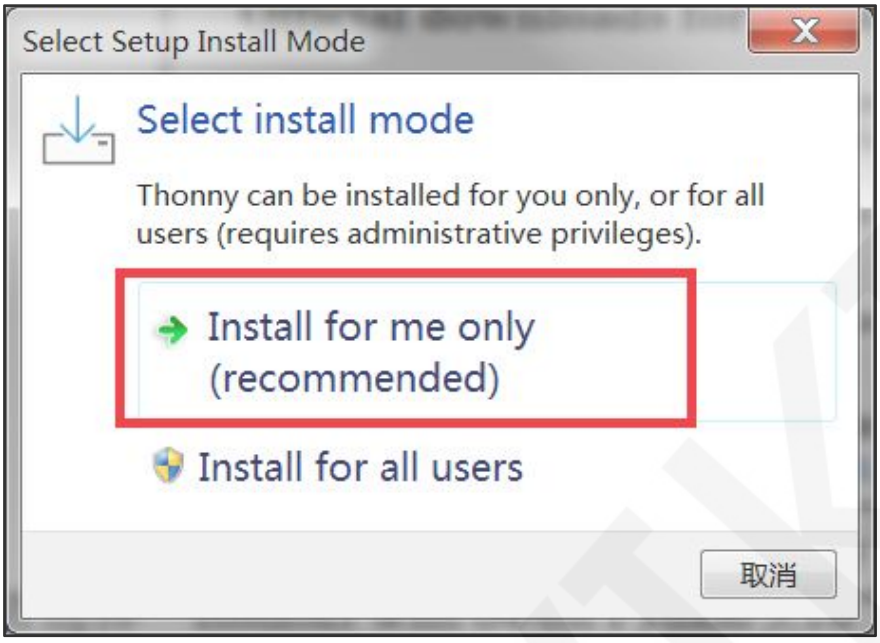

Figure 2.2 Thonny software installation mode selection

Then enter the Welcome to use Thonny interface, directly click the "**Next**" button, as shown below:

Figure 2.3 Thonny software welcome to the interface

Then enter the license agreement interface, select "**I accept the agreement**", and click "**Next**" button, as shown in the picture below:

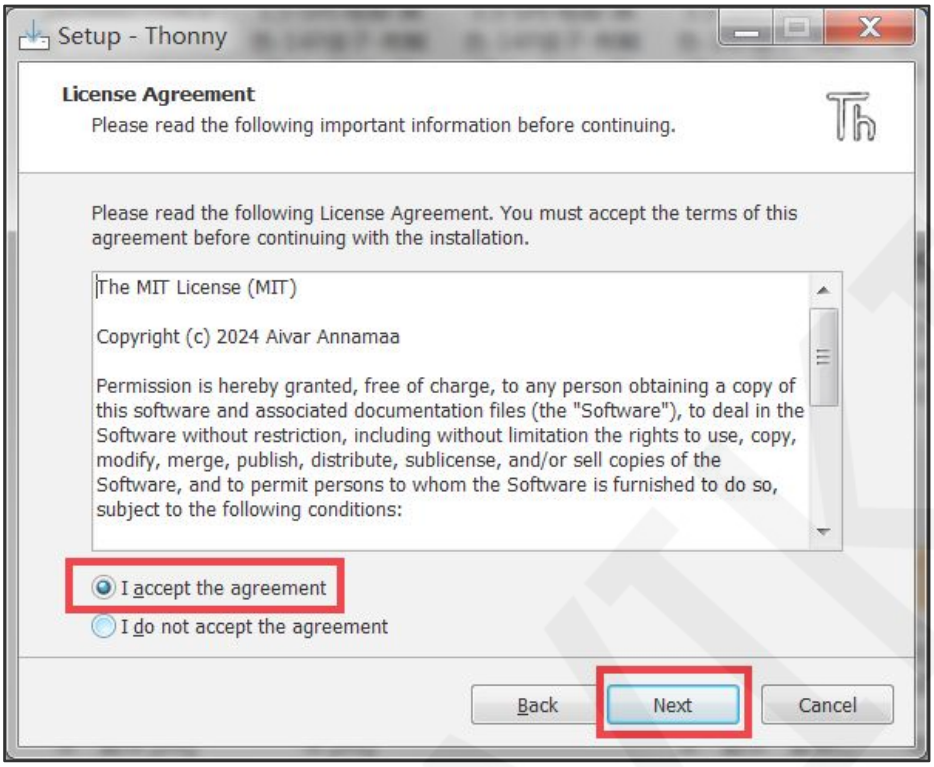

Figure 2.4 Thonny Software license agreement selection

Next, enter the interface of selecting the installation directory, click "**Browse...**" Select the installation directory (you can also use the default directory) and click the "**Next**" button, as shown below:

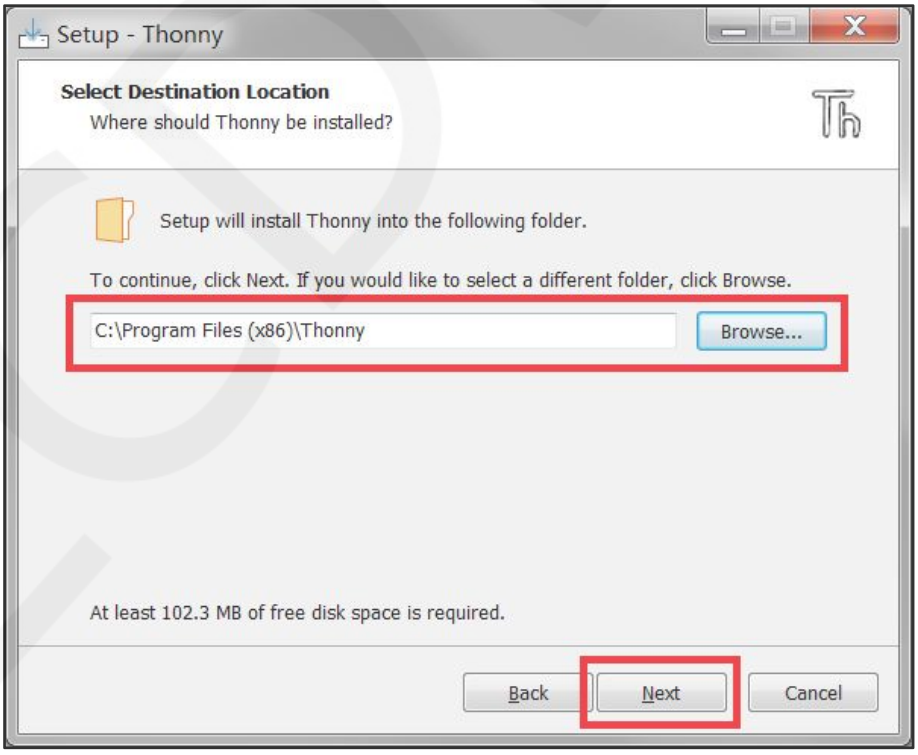

Figure 2.5 Thonny Software installation directory Select

Next, enter the Start menu folder selection interface, this folder is used to store the shortcut icon in the Start menu bar, click "**Browse...**" Select the button (default folder can also be used), then click the "**Next**" button, as shown below:

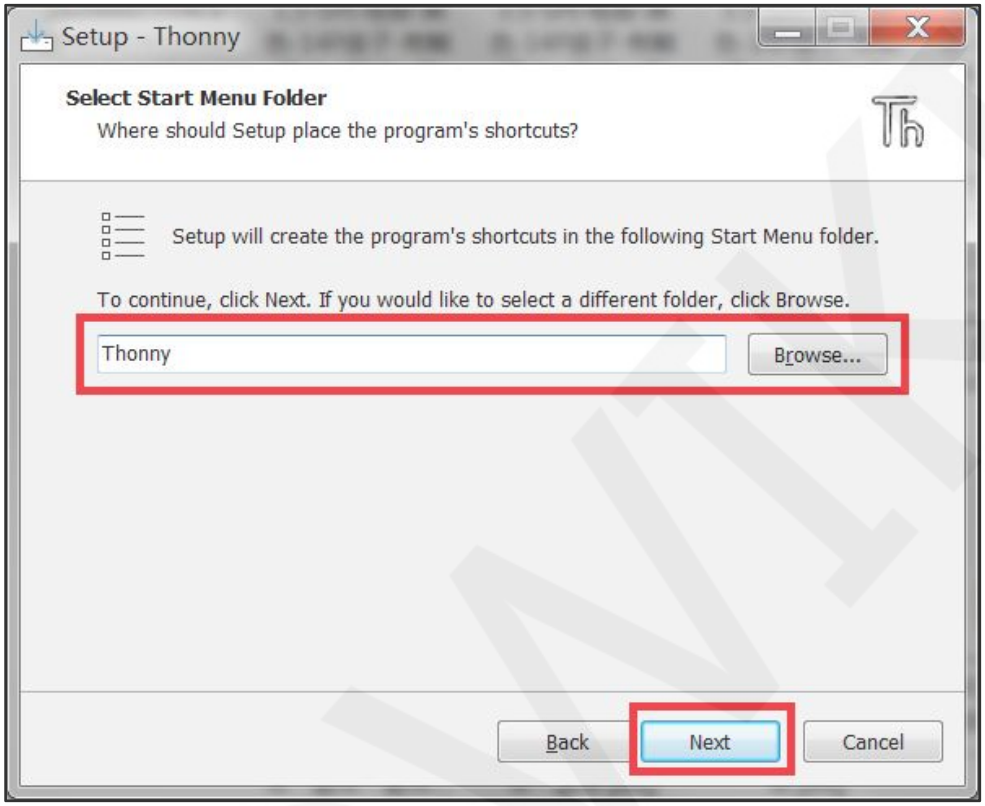

Figure 2.6 Thonny software Start menu directory selection

Next, go to the Select whether to Create desktop icon interface. If you select "**Create Desktop icon**", the desktop icon will be created. If you do not select it, the desktop icon will not be created. In order to facilitate opening, generally select the check box, and then click the "Next" button, as shown in the following picture:

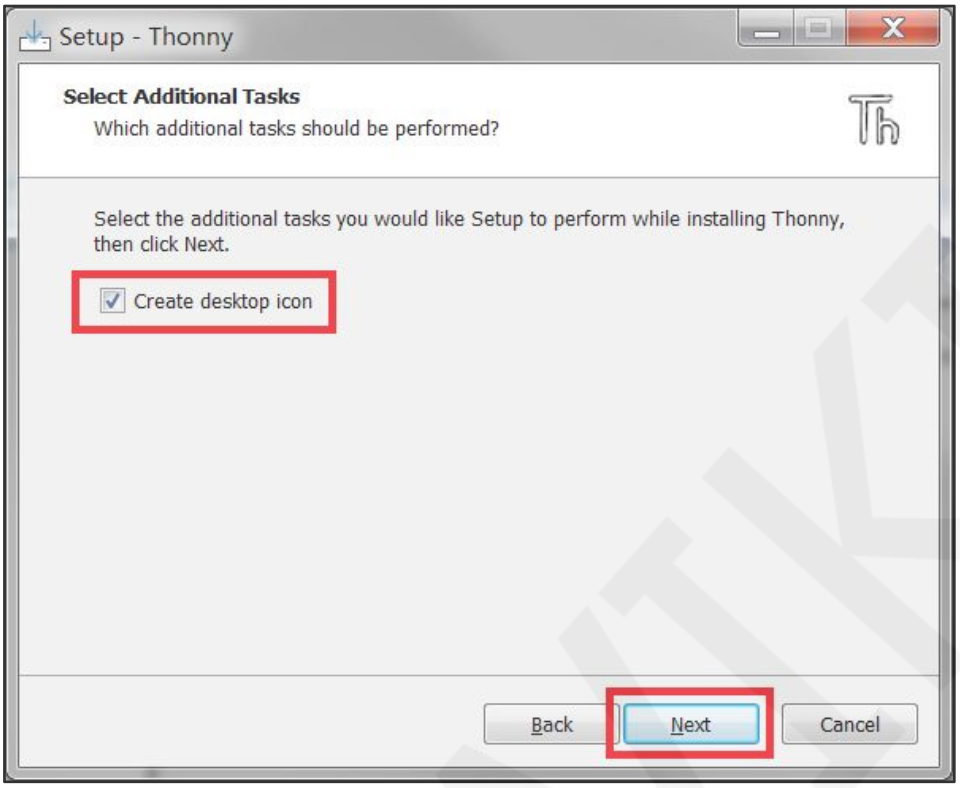

Figure 2.7 Thonny software creates desktop icon selection

Then enter the installation preparation interface, click the "**Install**" button to install, as shown in the picture below:

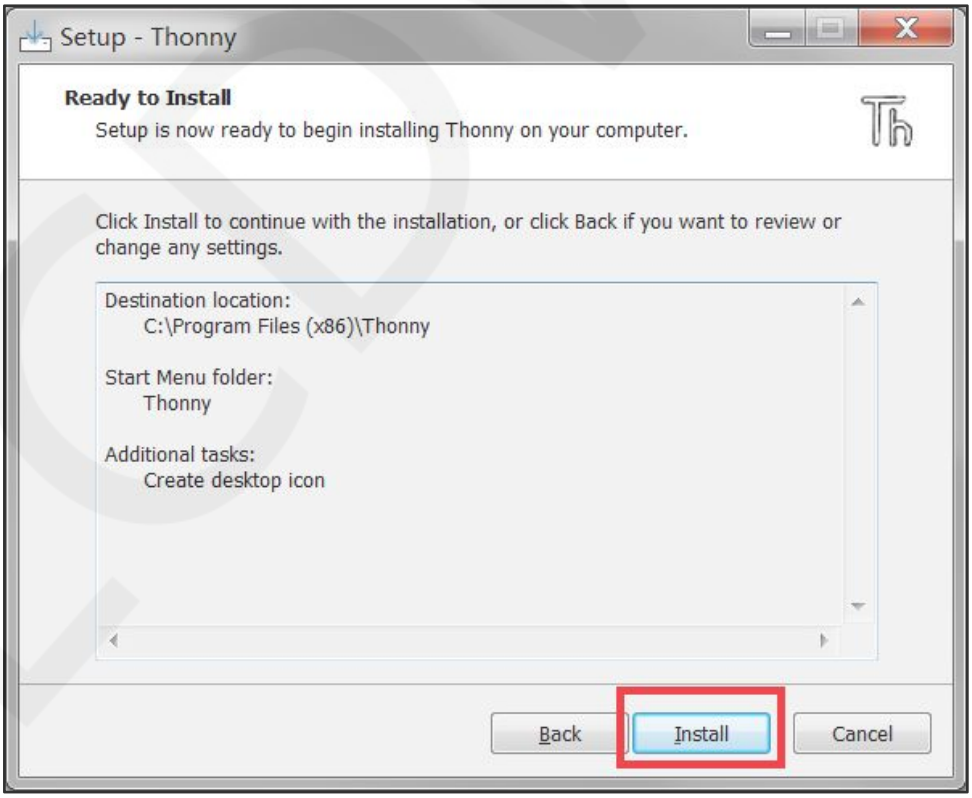

Figure 2.8 Prepare to install the Thonny software

Next, enter the software installation stage. After the progress bar is completed, the software will be installed, as shown in the figure below:

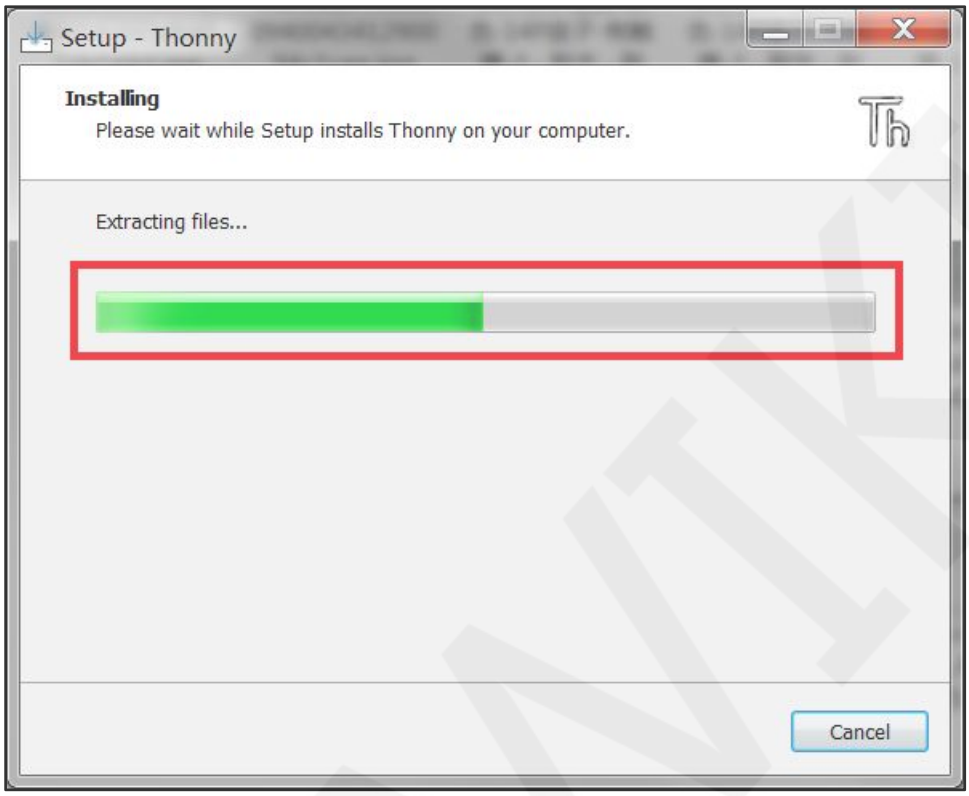

Figure 2.9 Thonny software installation

Next, enter the software installation completion interface, and click "**Finish**" button to close the interface, as shown in the picture below:

Figure 2.10 The Thonny software is installed

Through the above steps, you can see that the Thonny shortcut icon is generated on both the desktop and the Start menu bar. Click the icon to open the Thonny software (when it is run for the first time, it will prompt you to select language and initialization Settings, here you can select Chinese and standard Settings), input the following code in the editor bar, and then click the **Run** icon, you can see the content output in the **Shell** bar, as shown in the figure below. At this point, installing the Thonny software under Windows is complete.

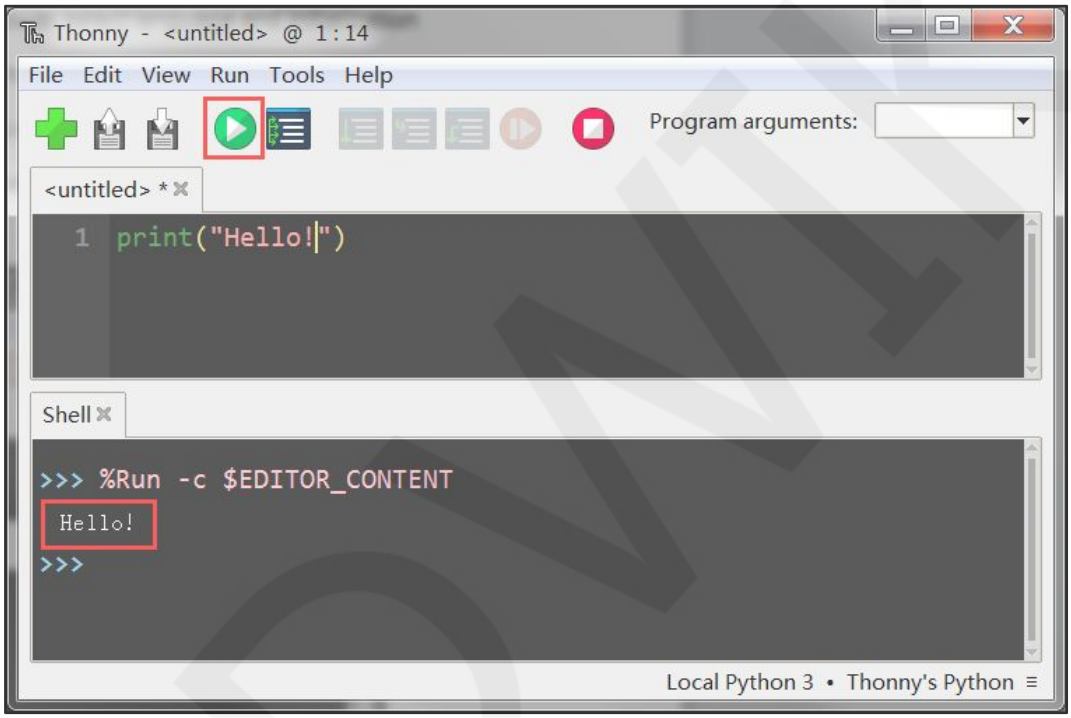

Figure 2.11 Thonny software is running properly

3. Thonny Software Introduction

Thonny software is easy to use, program code editing, automatic completion, debugging, compilation, upload, installation and other features, the main interface is shown as follows:

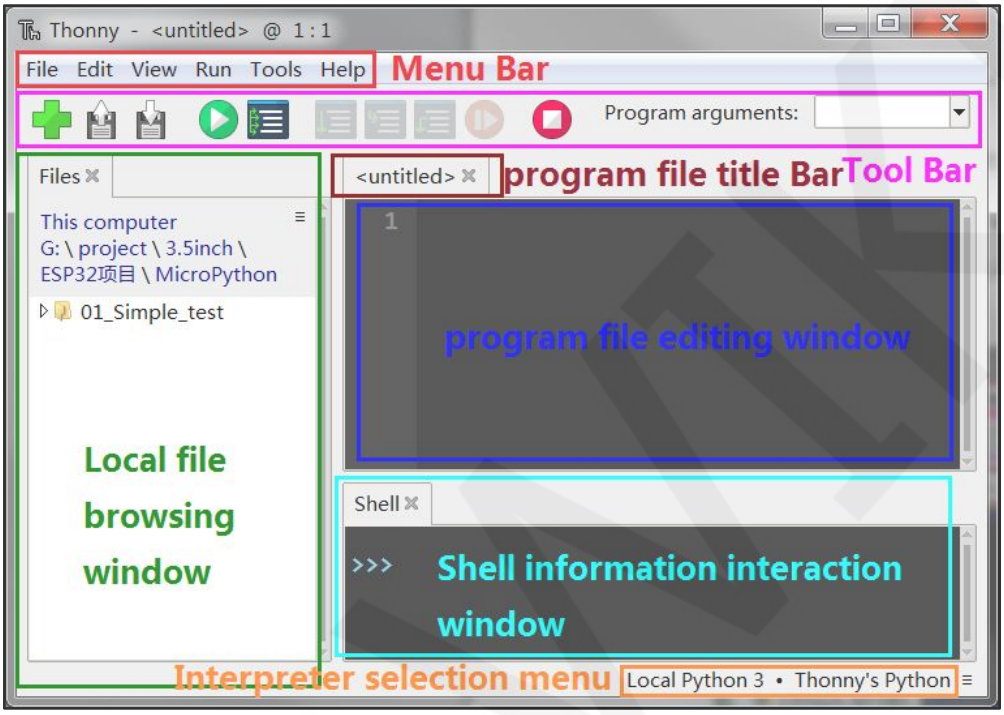

Figure 3.1 Thonny Main Interface

Menu bar: The main setting options of Thonny, most of the Settings are in this.

Tool bar: Some common operation buttons, easy to operate (can also be performed in the menu bar)

Local File Browse window: View Python files saved on your computer

Program file title bar: Display Python file name, display multiple files, easy to switch.

Program File editing window: Edit Python files.

Shell information interaction window: View the results of Python program running, while supporting the input of commands.

Interpreter selection menu: Set the Python interpreter.

3.1. Menu bar

3.1.1. File menu

The operations in the file menu mainly process files, including creating files,

opening existing files, opening recently operated files, closing current or all files, saving current or all files, saving files as, saving files as copies, moving or renaming files, printing files, and closing Thonny software functions. The file menu bar interface is shown below:

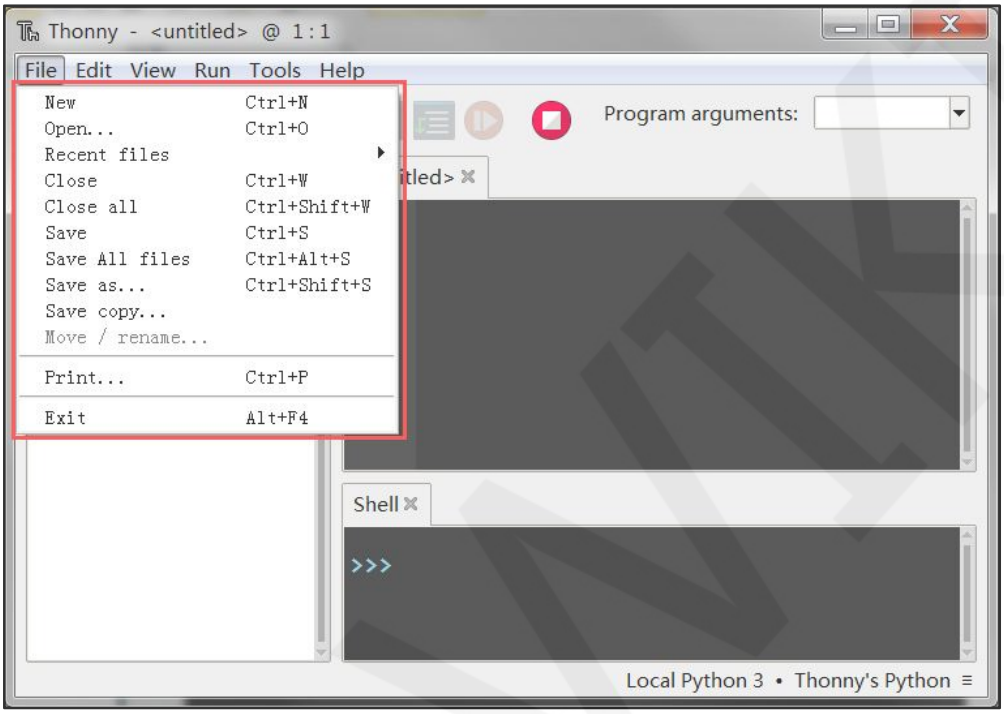

Figure 3.2 Thonny file menu bar

3.1.2. Edit menu

The editing menu of Menu bar provides the operation of editing files, including undo, redo, cut, copy, paste, select all, code line indentation adjustment, code comment adjustment, jump to the specified code line, code automatic completion, code parameter display, find and replace, empty Shell information and other operations. The editing menu interface is shown as follows:

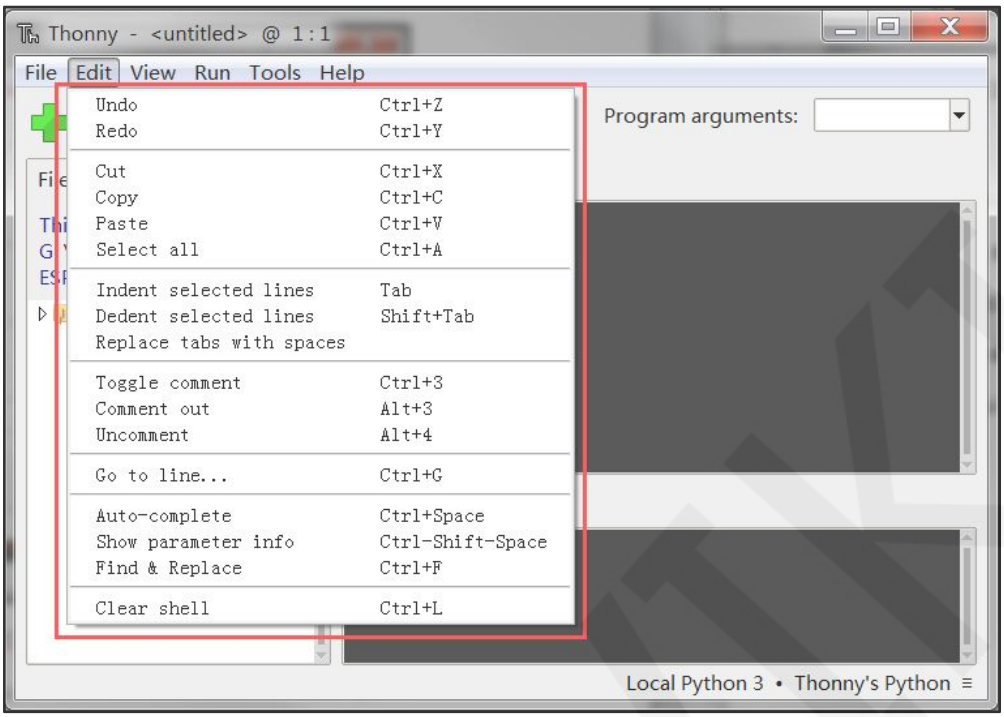

Figure 3.3 Thonny Edit menu bar

3.1.3. View menu

The View menu of Menu bar provides the operation of closing or opening a view window, Thonny provides more than a dozen information display Windows, information display is very intuitive. Click the corresponding window name. If "√" appears in front of the name, the window is open. Click again to close the window. In addition, the interface font resizing function and editor /Shell window switching function are provided. The view menu interface is as follows:

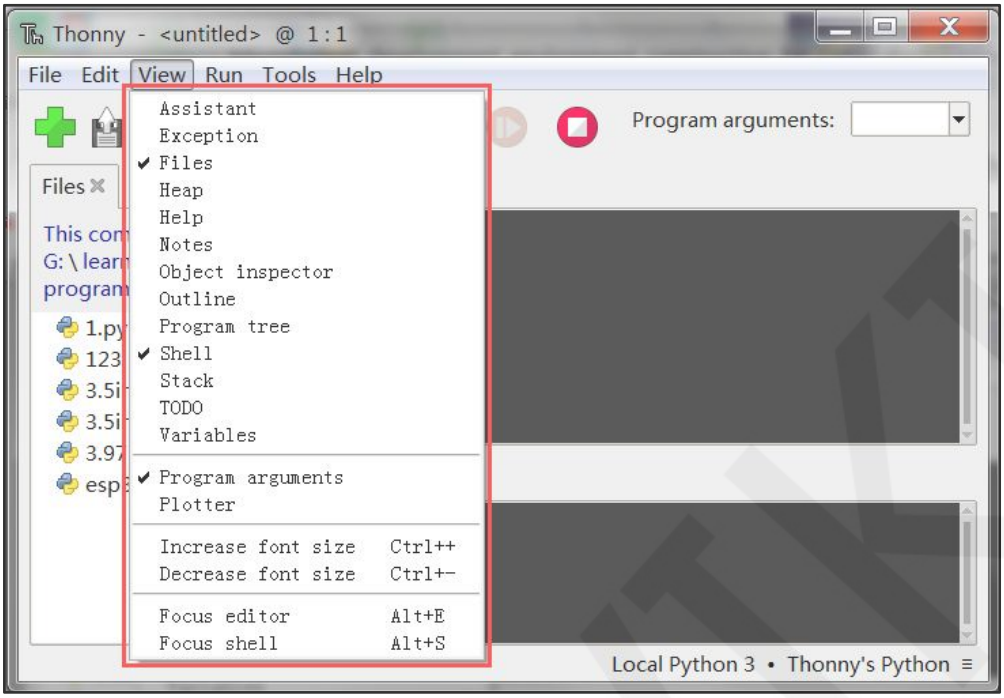

Figure 3.4 Thony View menu bar

3.1.4. Run Menu

The Run menu of Menu Bar provides operations such as configuring the interpreter, running Python programs, debugging Python programs, stopping/restarting back-end processes, interrupting execution, and sending EOF/ soft restart. There are three debugging methods: nicer, Faster, birdseye, and it can also run the program in a single step, so that users can view the process of running the program. The running menu interface is shown as follows:

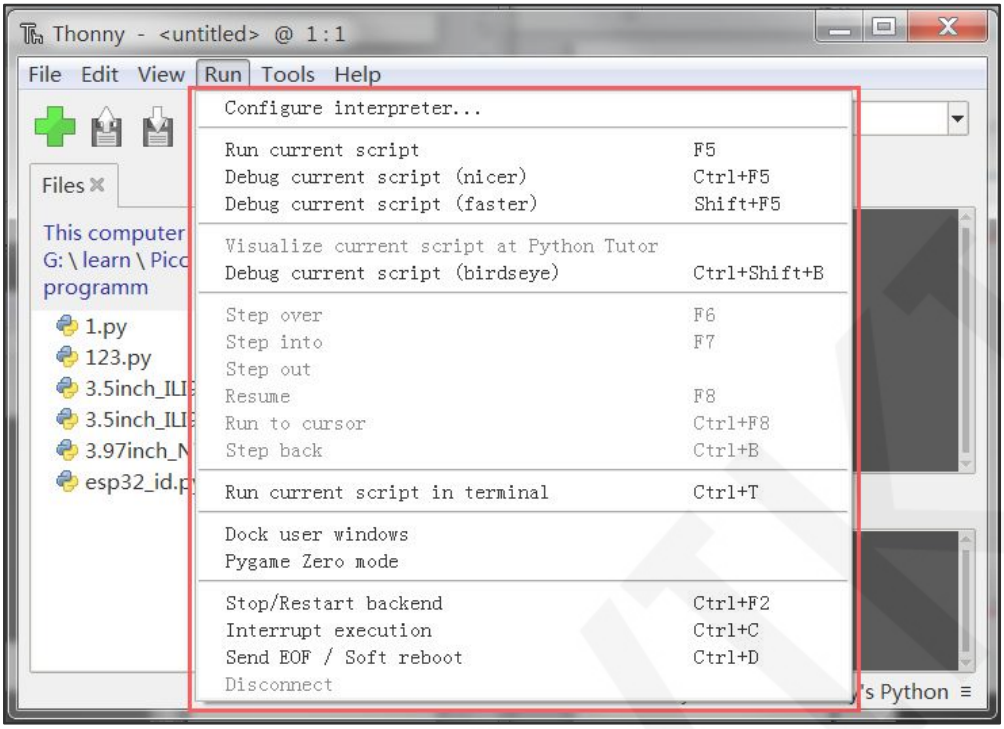

Figure 3.5 Thonny Run menu bar

3.1.5. Tools Menu

The tool menu bar mainly provides software package management, opening the system Shell window, opening the Thonny installation directory and data directory, plug-in management, option setting and other operations. The tool menu interface is shown as follows:

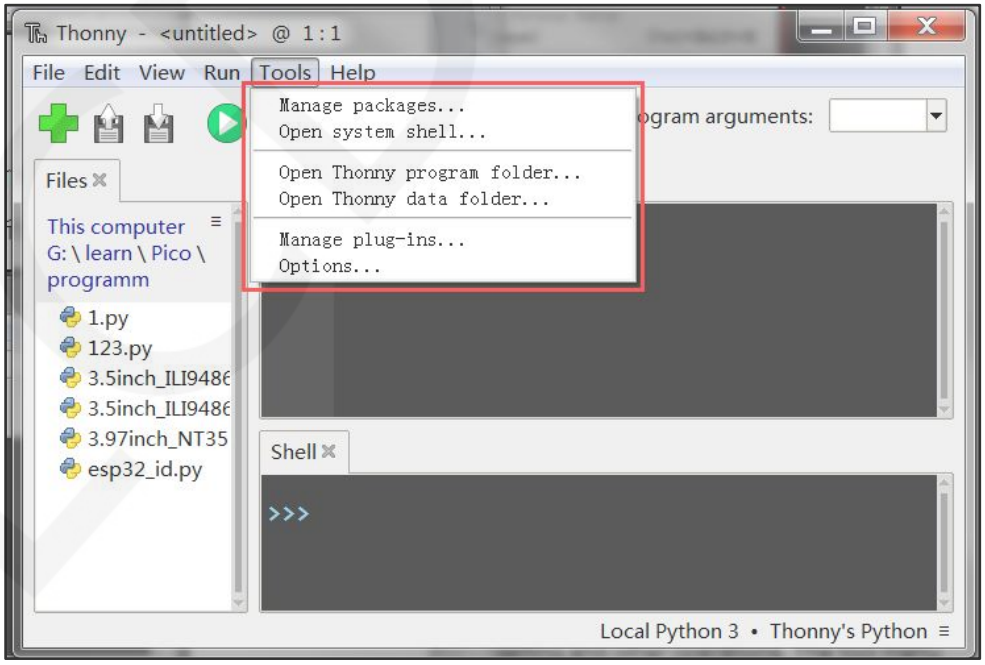

Figure 3.6 Thonny Tools menu

Click on "**Manage packages...**" from the Tools menu. Option to enter the software package installation management page. The installed software packages are listed on the left. You can click the corresponding package name to view details. The interface provides three ways to install the software package: the first is to enter the package name from the search bar, click the "**Search on PyPI**" button to search, and then install according to the search result, the second is to install according to the package specified in the file, and the third is to install from the local file.

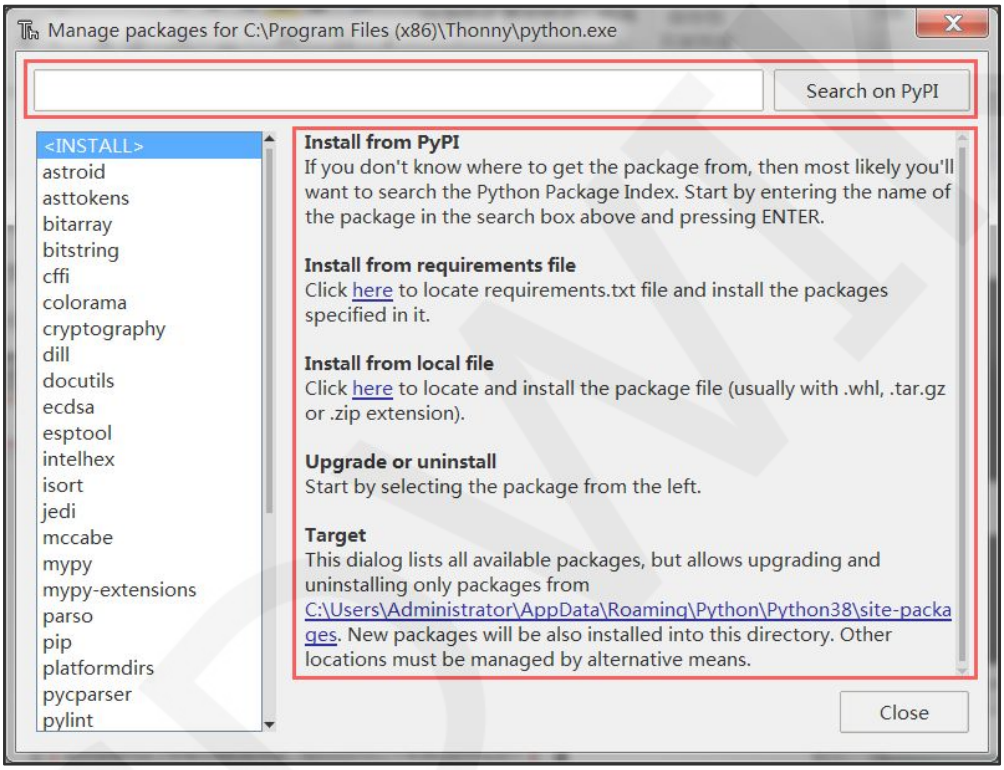

Figure 3.7 Thonny package management

Click on "**Manage plug-ins...**" from the Tools menu. Enter the plug-in installation management interface, enter the plug-in name from the search bar, click the "**Search on PyPI**" button to search, and then install according to the search result.

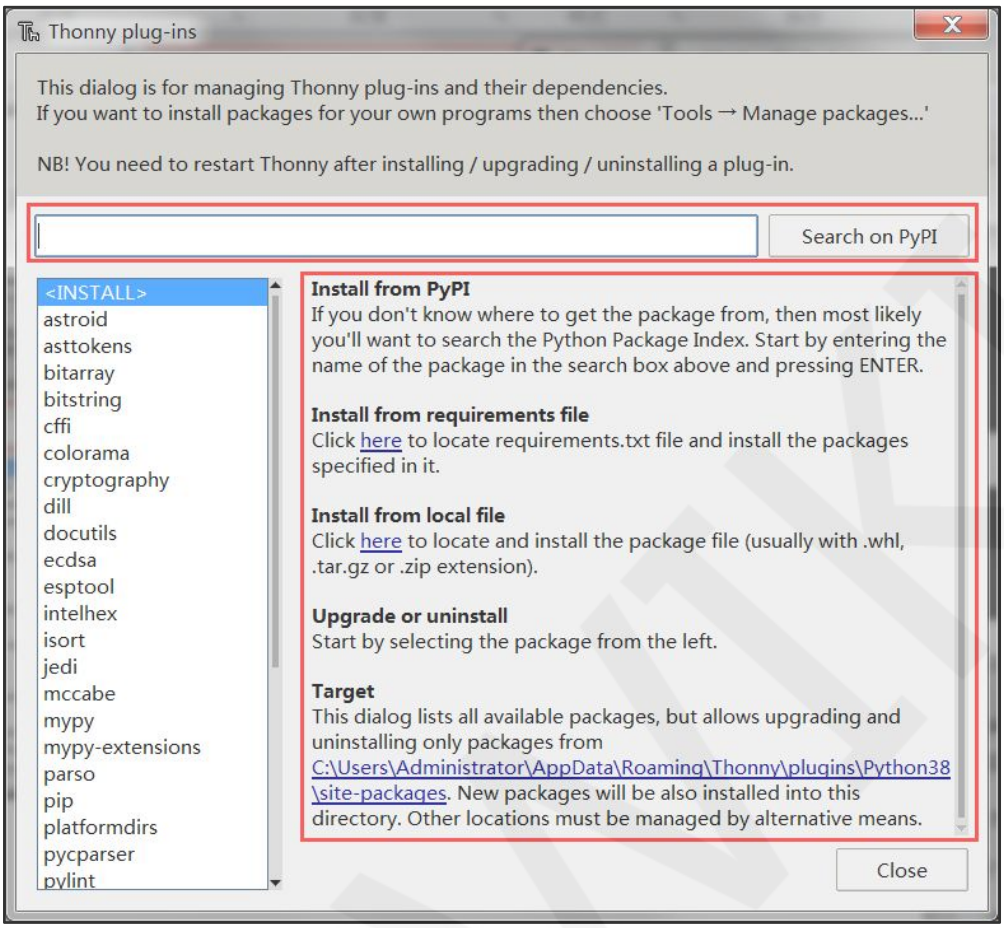

Figure 3.8 Thonny plugin management

Click on "**Options...**" from the Tools menu Option to enter the Thonny option screen. The first is the general setting interface, which provides some Settings when running the program, as well as interface language Settings, UI-related Settings and environment variable Settings, which need to be restarted after the setting is completed. The interface is shown as follows:

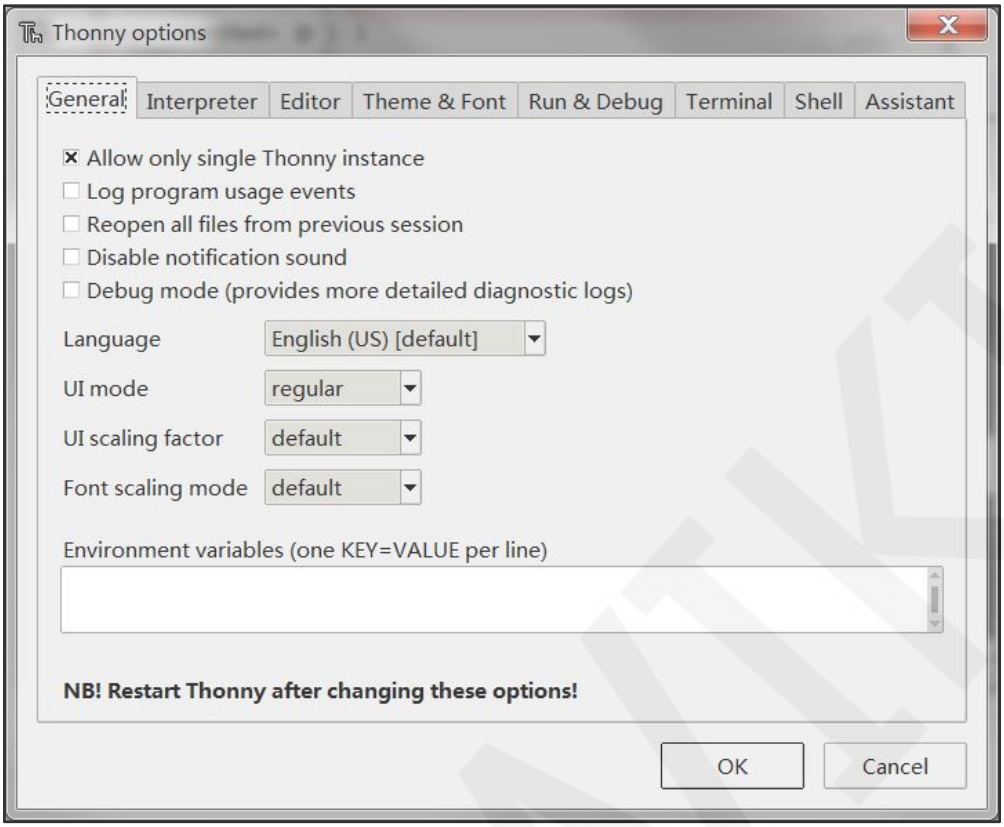

Figure 3.9 Thonny General Settings

Click the "**Interpreter**" button to enter the Python interpreter setting interface. The Python interpreter is the core of Python program operation and is responsible for interpreting Python programs and converting them into machine language so that computers can execute them. Different hardware platforms need to select the corresponding Python interpreter. Otherwise, an exception may occur when running Python. The interface is as shown below (here the interpreter Settings are the same as those in the run menu) :

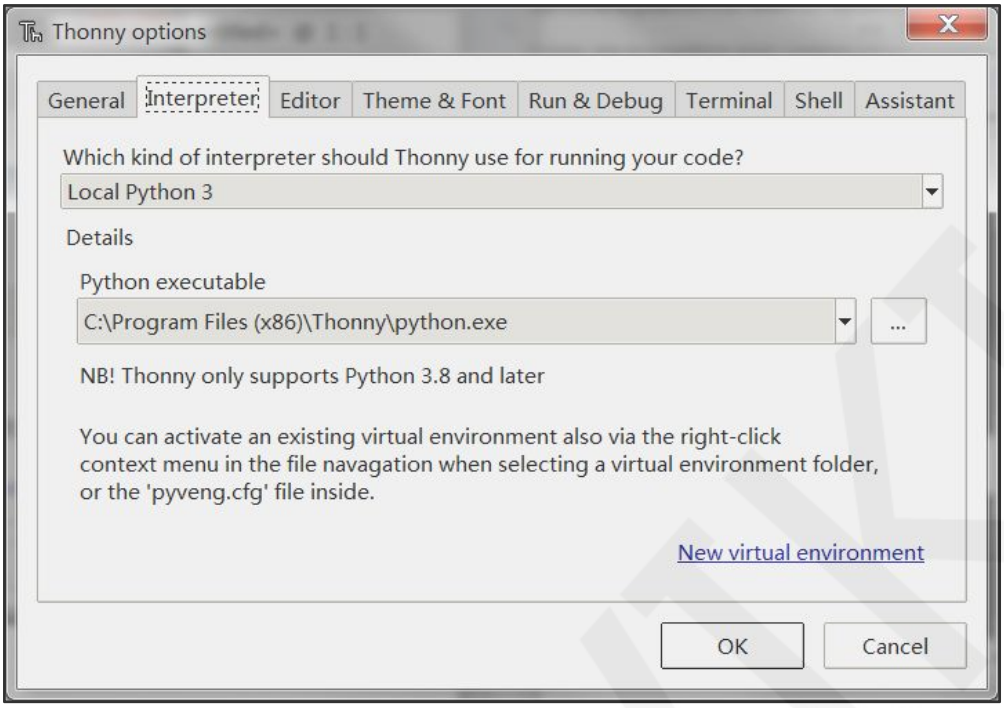

Figure 3.10 Thonny Interpreter Settings

Click the "**Editor**" button to enter the Python editor setting interface, which mainly displays and sets the input content of the code editor, including highlighting, automatic prompt completion and other Settings, as shown in the following picture:

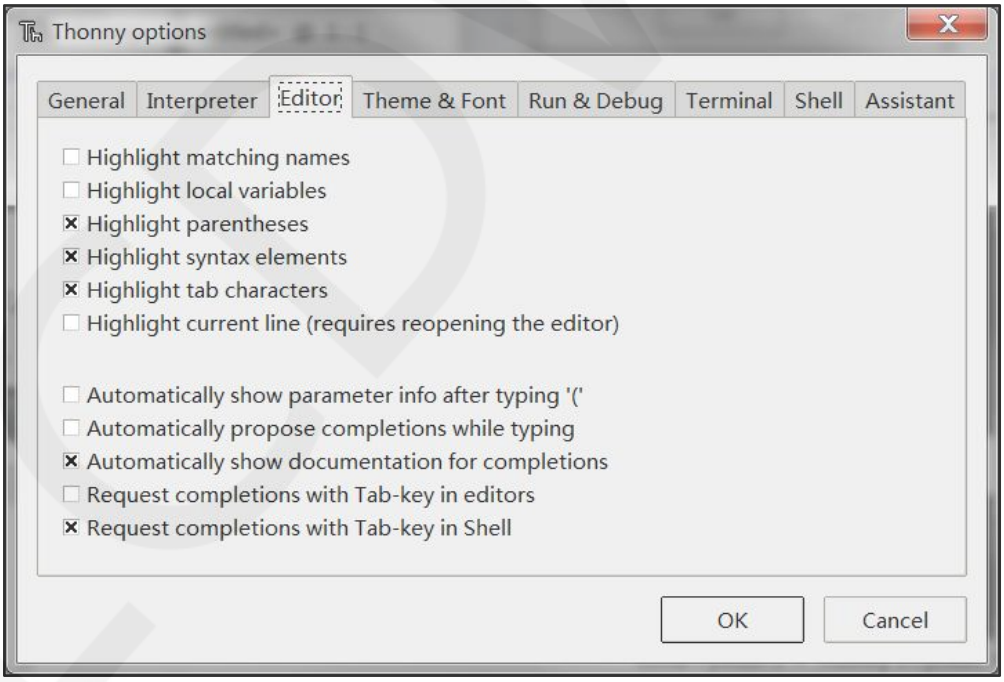

Figure 3.11 Thonny Editor Settings

Click the "**Theme & Font**" button to enter the software theme and font setting interface, which provides users with personalized choices as long as the theme style,

font style and size displayed by the software are set. The interface is shown as follows:

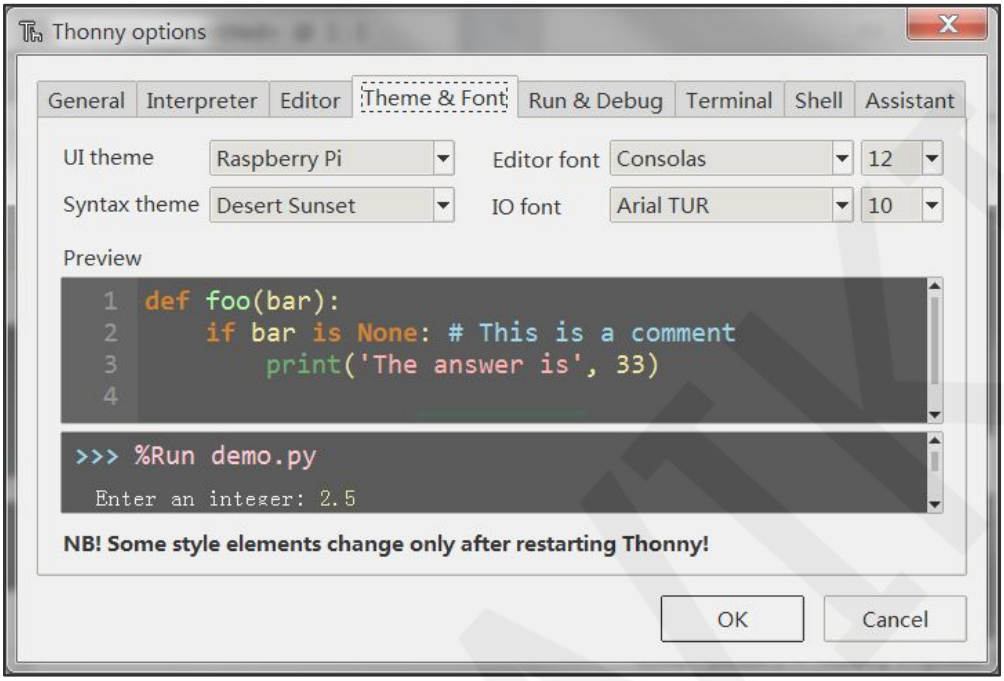

Figure 3.12 Thonny Theme&Font Settings

Click the "**Run & Debug**" button to enter the program run and debug setting interface. The interface is shown as follows:

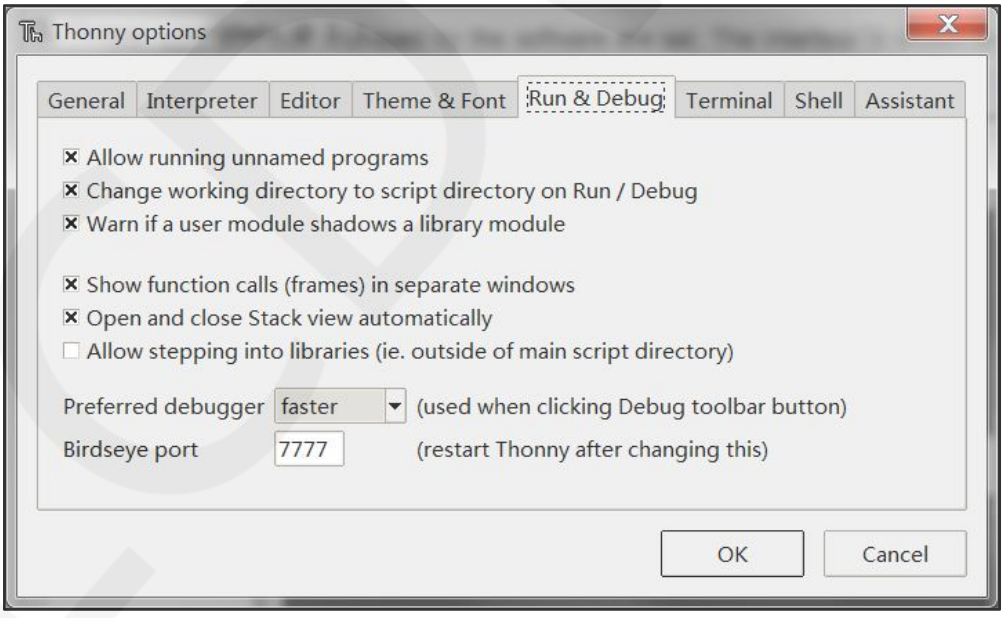

Figure 3.13 Thonny Run&Debug Settings

Click the "**Terminal**" button to enter the terminal setting interface when the program is executed. The interface is shown as follows:

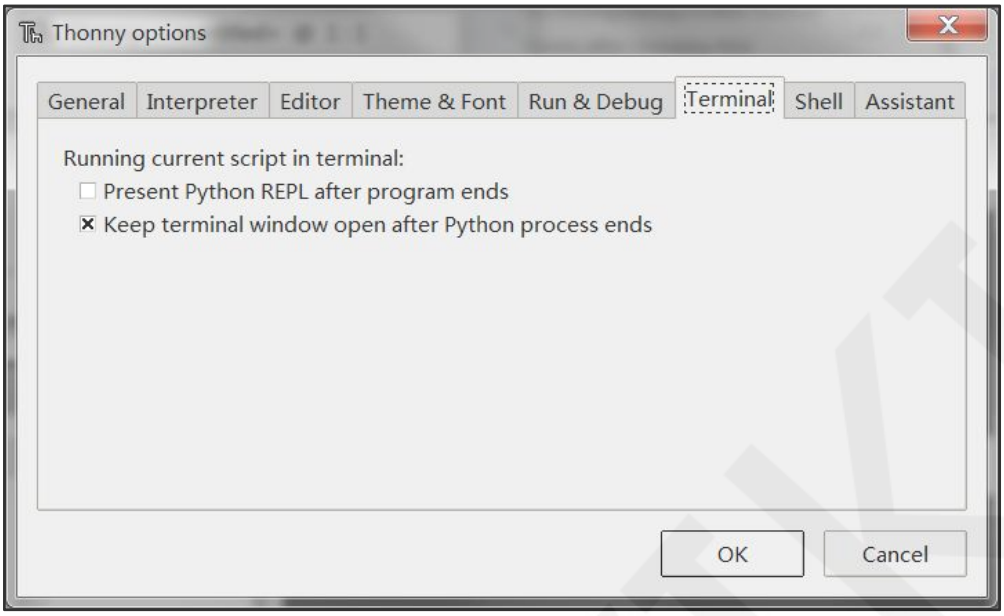

Figure 3.14 Thonny Terminal Settings

Click the "**Shell**" button to enter the Shell window setting interface, which mainly sets the Shell display content, including whether to empty the Shell display content and the maximum displayed line setting, as shown in the following picture.

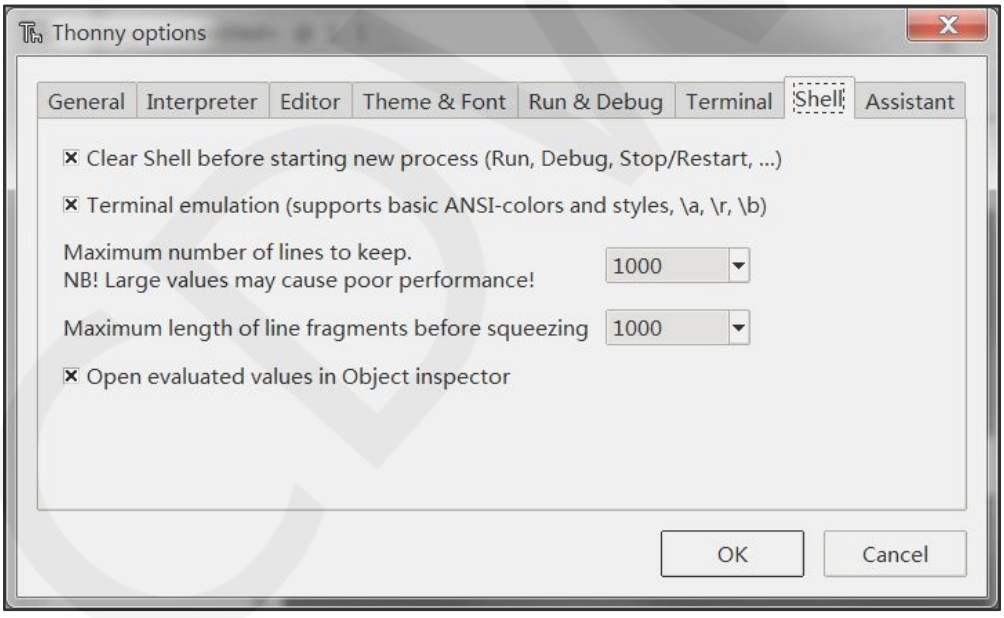

Figure 3.15 Thonny Shell Settings

Click the "**Assistant**" button to enter the assistant setting interface, which mainly checks the running status of the code, as shown in the picture below. Once the appropriate options are set, click the "**OK**" button.

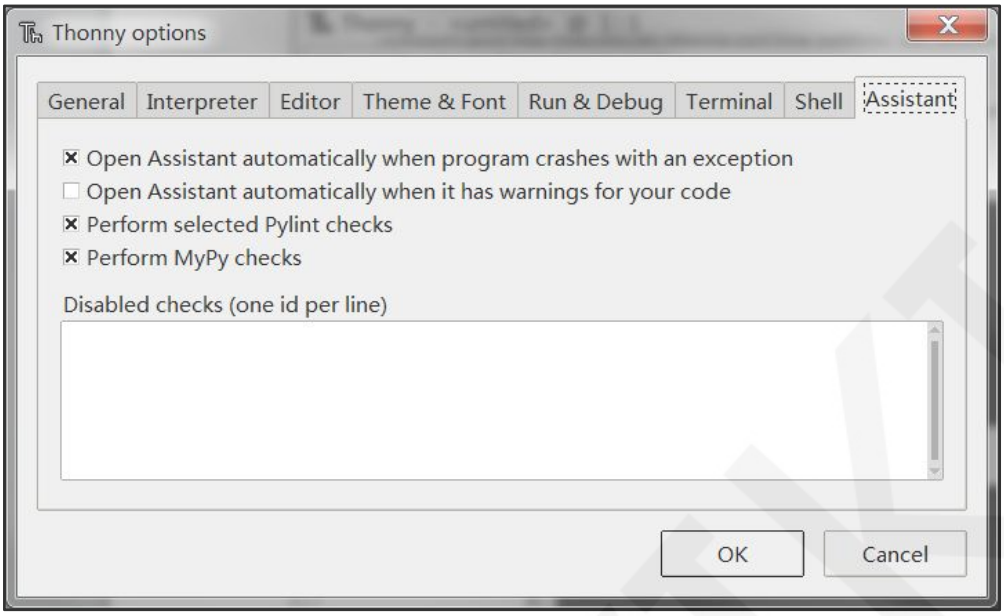

Figure 3.16 Thonny Assistant Settings

3.1.6. Help Menu

The help menu bar mainly provides some software use help, view historical versions, feedback software problems and a brief introduction of Thonny software. The help menu interface is shown as follows:

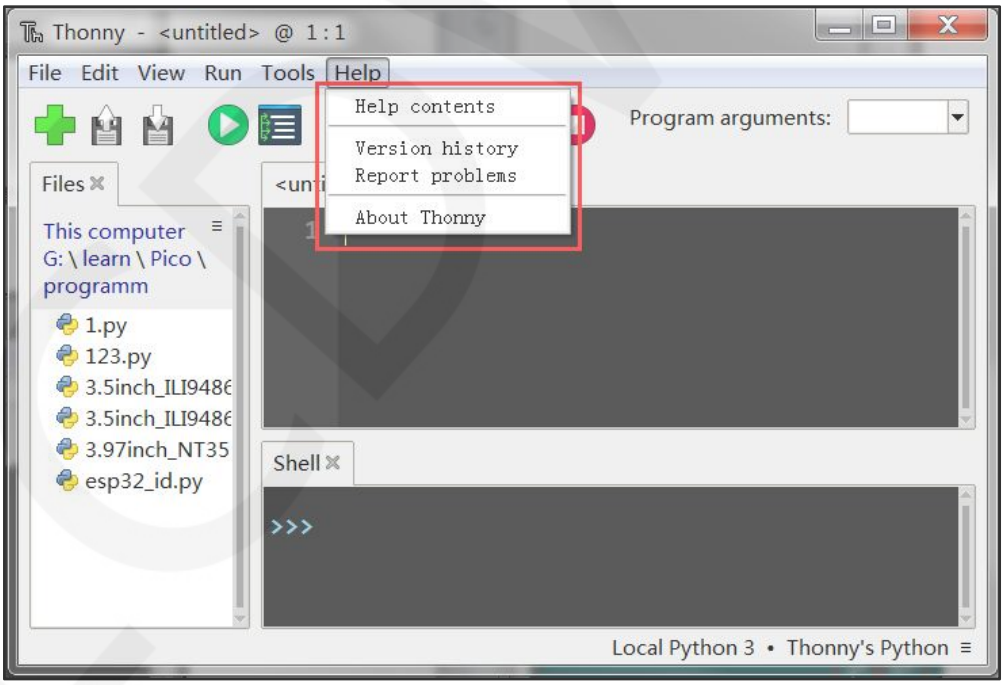

Figure 3.17 Thonny Help Menu

3.2. Tool bar

The toolbar provides shortcut buttons for common operations that are easy to perform (these operations can also be performed in the menu bar). Shown as follows:

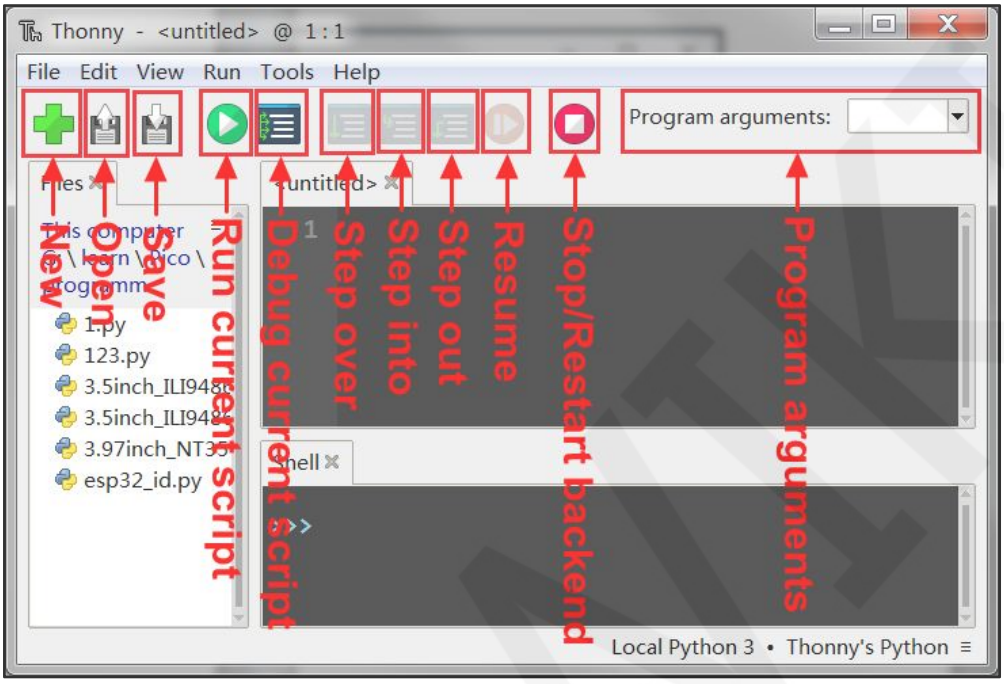

Figure 3.18 Thonny Tool bar

New: Recreate a Python file.

Open: Open an existing Python file.

Save: Save the currently open Python file.

Run current script: Execute the currently open Python file.

Debug current script: Debug the currently open Python file to see how the program is running.

Step over: Run a program function at each step.

Step into: Run one line of code at each step (single step debugging).

Step out: The entire program runs until the end of each step.

Resume: When debugging is paused, perform this operation and run directly to the end of the program.

The preceding four operations can be performed only after entering the debugging phase.

Stop/Restart backend: Stop or restart the Shell back-end process.

Program atguments: Displays the parameters in the program.

4. Download and burn ESP32 MicroPython firmware

To get started with MicroPython on the ESP32, the preferred option is to burn the ESP32 MicroPython firmware. ESP32 MicroPython firmware is a bridge between the underlying hardware and the upper application layer, it encapsulates the operation method of the underlying hardware, and then provides interfaces and methods for the upper application layer to call, which greatly reduces the difficulty of development.

4.1. Install the USB-to-serial IC driver

Before downloading the ESP32 MicroPython firmware, you need to install the USB-to-serial IC driver. If no, the computer cannot identify the serial port. Drivers need to be installed for different USB-to-serial IC types. Here the module uses the CH340C USB-to-serial port driver IC, so to install the CH340 driver, perform the following steps (if already installed, skip the following steps) :

A. Find the **USB-SERIAL_CH340.zip** package in the“**7-工具软件_Tool_software**” folder and decompress it.

B. go to the folder after decompression, double-click "**CH341SER.EXE**" executable program, pop up the installation window, and then click "**Install**" button to continue the installation, as shown in the following picture:

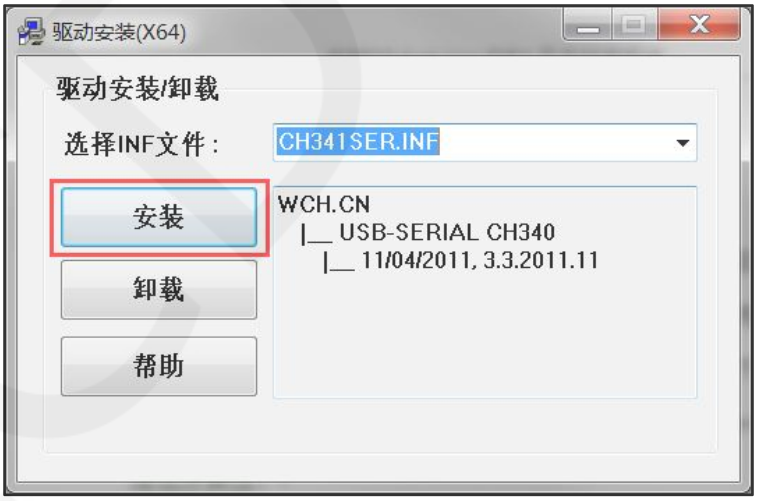

Figure 4.1 Install the CH340C driver

C. After the installation is successful, click the window OK button to exit. Connect the computer USB to the ESP32 module to power on, and then enter the computer device manager, you can see that the identified CH340 port appears

under the port, as shown in the following picture:

Figure 4.2 Identifies the CH340 port

4.2. Download ESP32 MicroPython firmware

ESP32 MicroPython firmware can be downloaded directly from the official website.

Official download website: https://micropython.org/download/

After entering the download page, click the **esp32** option under "**Port**" or "**MCU**"

Figure 4.3 ESP32 MicroPython Firmware Download 1

Next go down to the "**ESP32/WROOM Espressif**" option, as shown below:

Figure 4.4 ESP32 MicroPython Firmware Download 2

Click the ESP32 option to go to the firmware download page, you can see that there are several ESP32 MicroPython firmware on the page, each firmware description is as follows:

Firmware: Used on most ESP32 devices with 4MB Flash

Firmware (Support for OTA): This command is used to create an OTA upgrade partition

Firmware (ESP32 Unicore): For modules containing a single-core ESP32 chip

Firmware (Support for SPIRAM / WROVER): For ESP32 modules with SPIRAM or WROVER modules

Firmware (ESP32 D2WD): For ESP32 devices with 2MB Flash

Here you use the dual-core ESP32 WROOM module with 4MB Flash and no SPIRAM, so select **Firmware** and click "**v1.23.0 (2024-06-02).bin**" to download the firmware, as shown below:

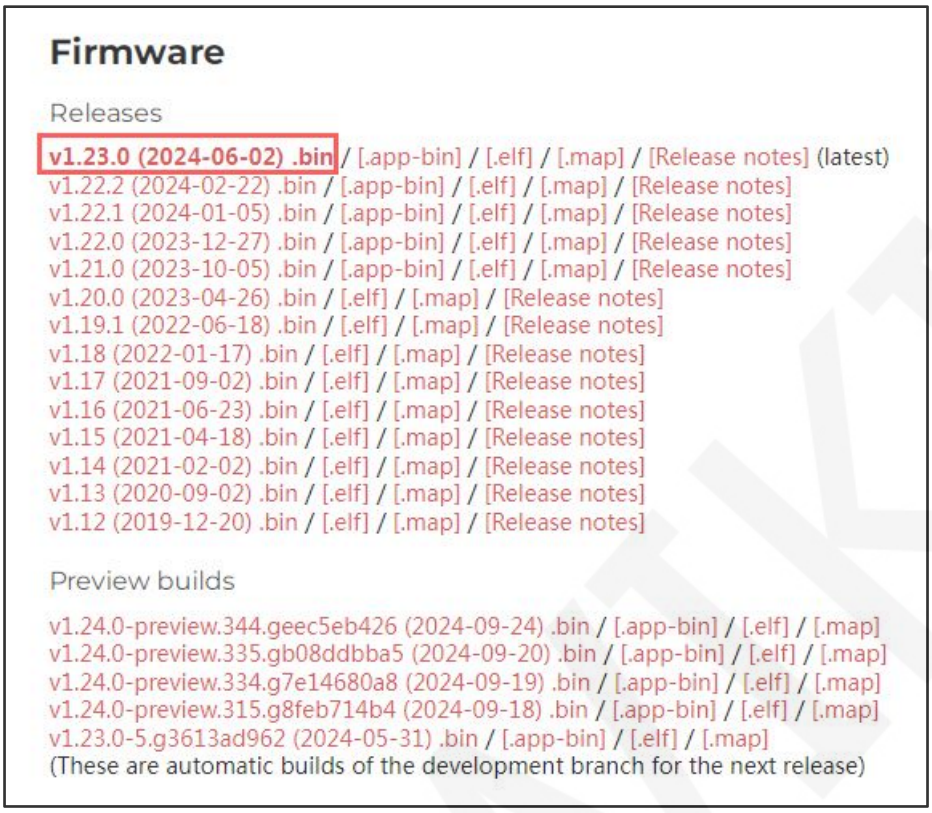

Figure 4.5 ESP32 MicroPython Firmware Download 3

Select the firmware storage directory and download it ,as shown below:

Figure 4.6 Select a directory to save the firmware and download it

4.3. Burn ESP32 MicroPython firmware

Once the ESP32 MicroPython firmware is downloaded, burn it.

Note: Before burning, the ESP32 module must be connected to the USB port of the computer and the COM port of the ESP32 module cannot be occupied by any program, otherwise the burning will fail.

There are two methods for burning: using Thonny software and using

flash_download_tool. The two methods are described in detail below:

4.3.1. Burn firmware using Thonny software

First, click the software menu bar to **Run** -> **Configure interpreter…**, or click the menu bar **Tools** -> **Options…** -> **Interpreter**, or click the button at the bottom of the Thonny software interface and select Configure Interpreter. Select the **MicroPython(ESP32)** interpreter in the interpreter interface, select the actual port number, and then click "**Install or update MicroPython (esptool)**", as shown below:

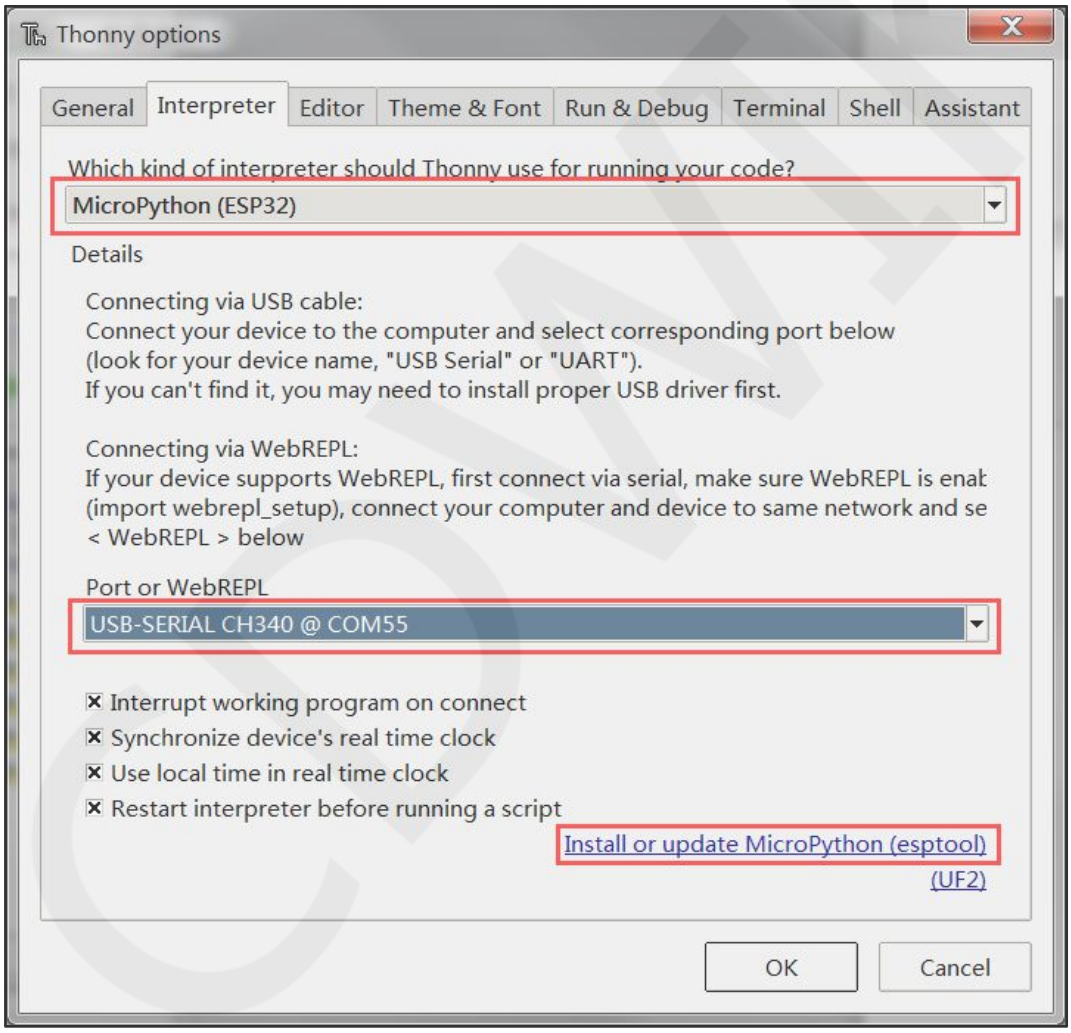

Figure 4.7 ESP32 MicroPython firmware burn 1

If you can't find esptool, click **Tools** -> **Manage Packages...** And **Tools** -> **Manage plug-ins...** Install esptool. As shown in the picture below:

If no error occurs, ignore this step.

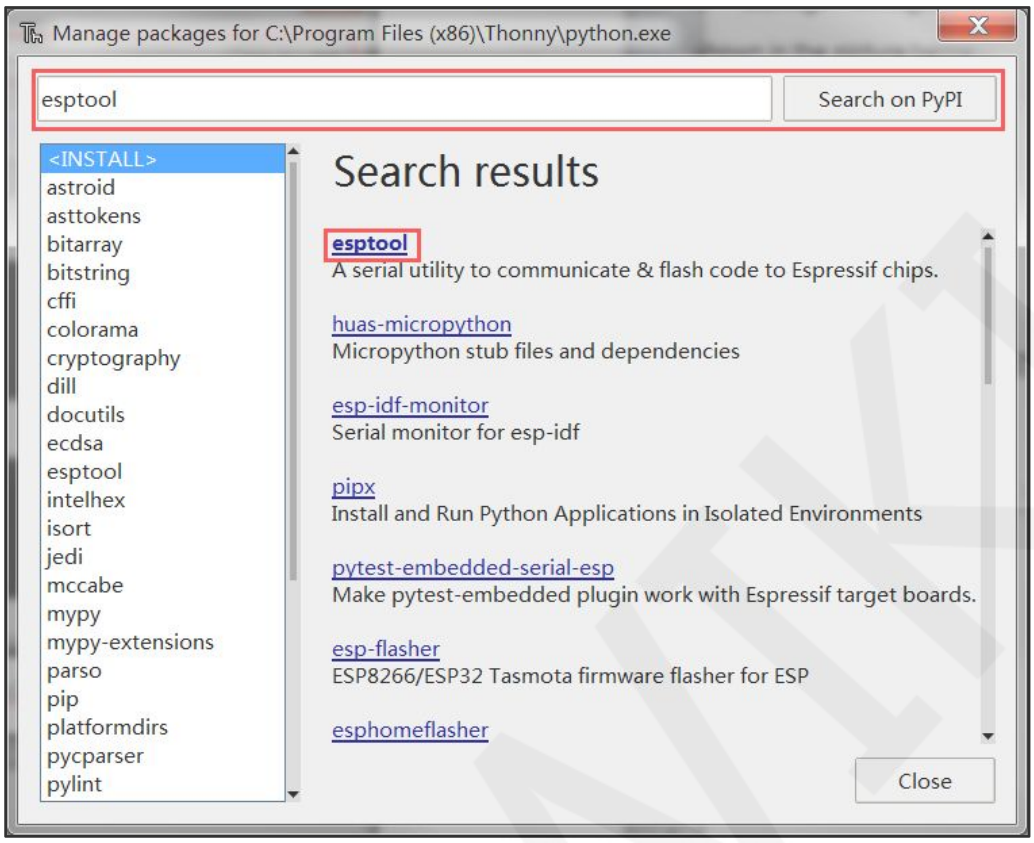

Figure 4.8 Install the esptool software package

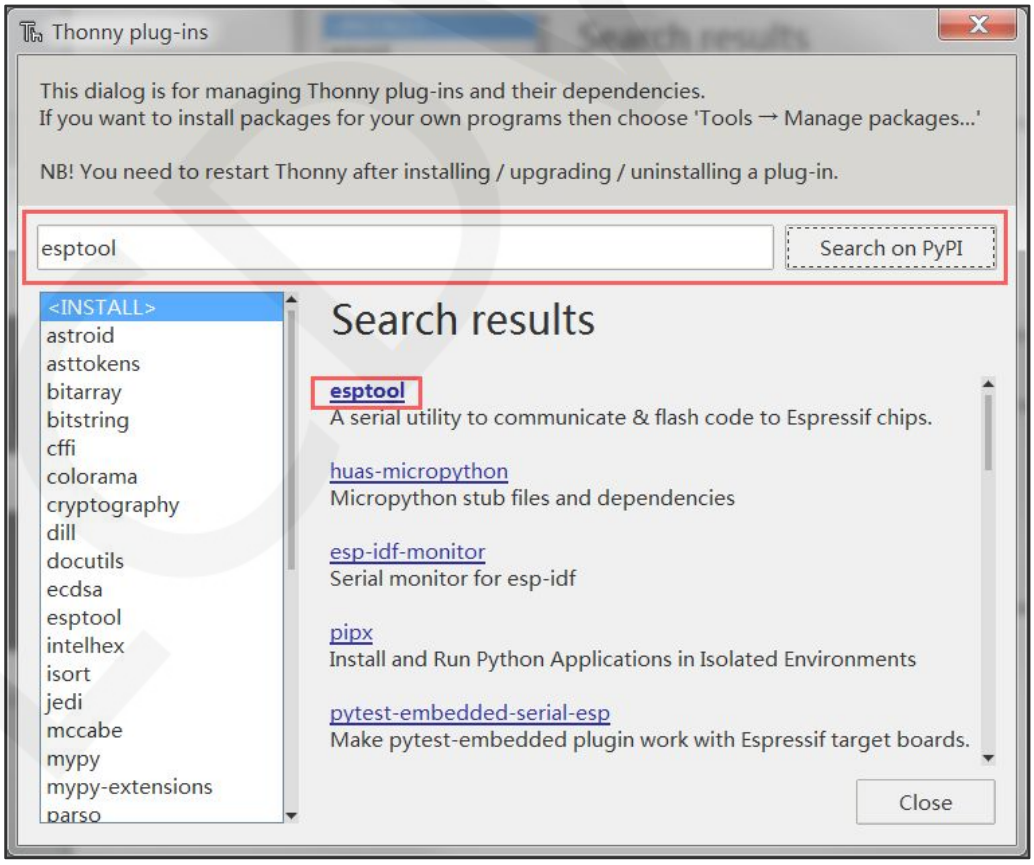

Figure 4.9 Install the esptool plug-in

If esptool is installed, go back to the **Configuration interpreter** screen and click “**Install or update MicroPython (esptool)**”.

There are two ways for Thonny software to burn ESP32 MicroPython firmware: local burning and online burning.

A. Local burning

Local burning means burning firmware files saved on the local computer to the ESP32 device.

After entering the burning interface, Select the correct port first, then click the bottom leftmost button, and click "Select local MicroPython image..." in the pop-up drop-down menu. Options, as shown below:

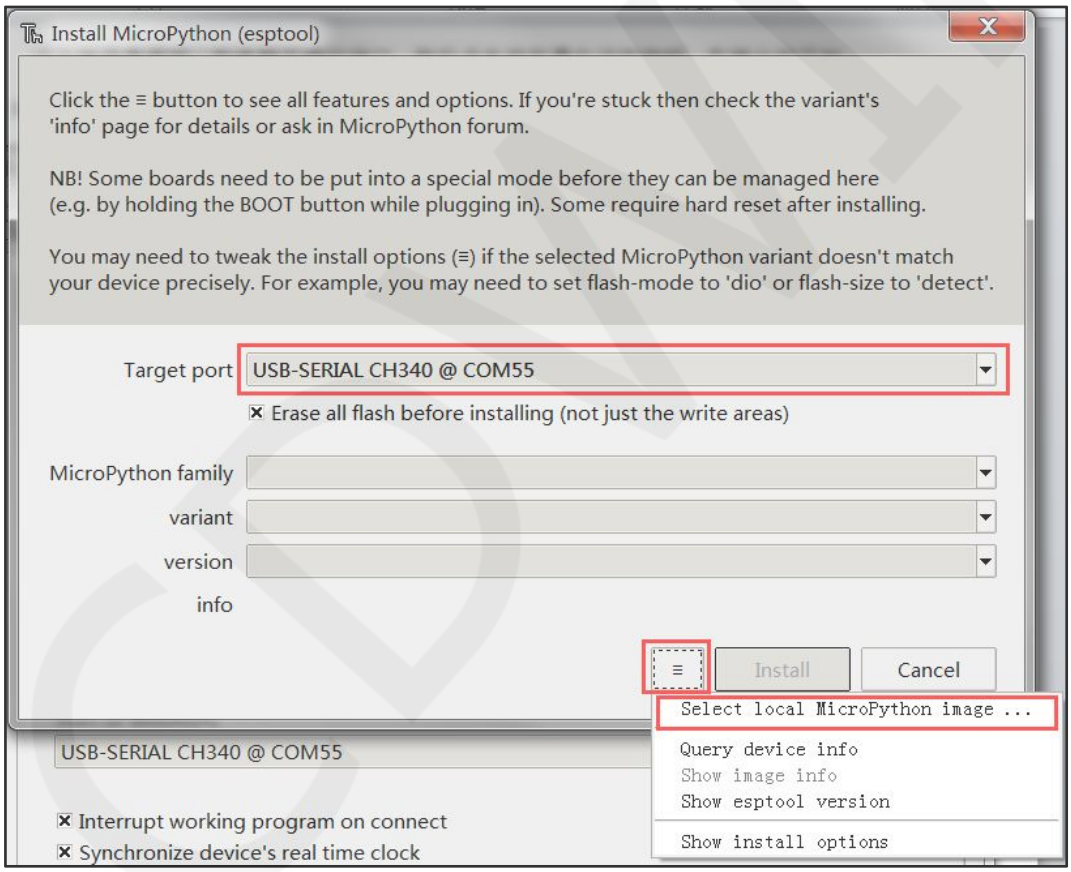

Figure 4.10 Select local burn firmware

Select the local firmware that needs to be burned in the pop-up directory, and the configuration information will be automatically filled at this time. If you do not need to perform more detailed configuration, click "**Install**" to burn the firmware. As shown in the picture below:

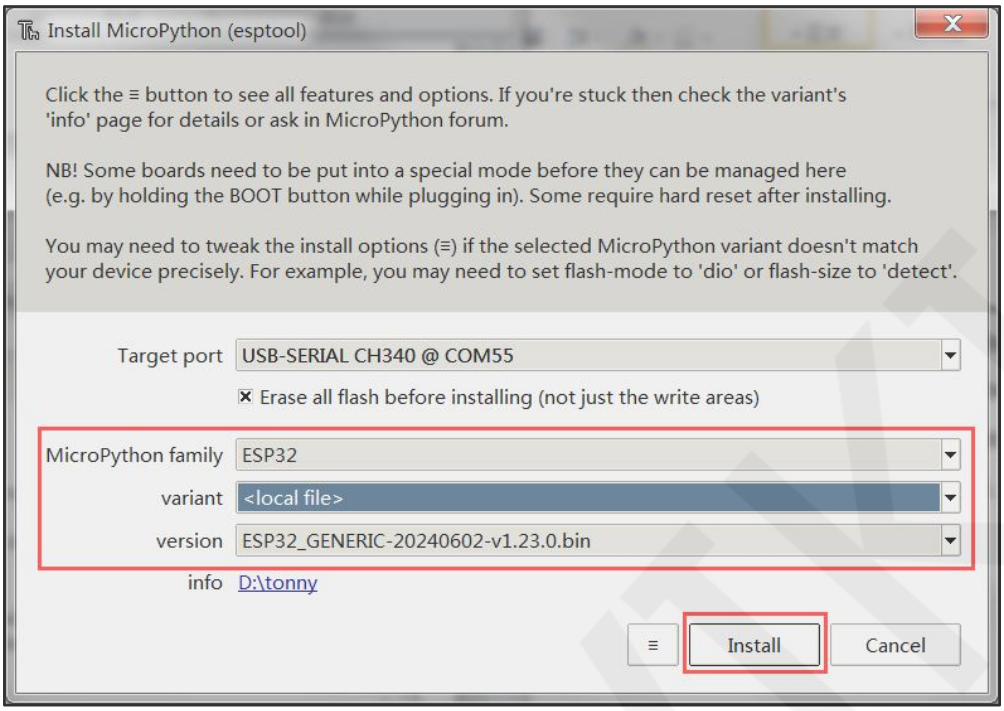

Figure 4.11 Burn the local firmware

If you need to perform more detailed configuration, click the leftmost button at the bottom first, and click "**Show install options**" in the pop-up drop-down menu, as shown below:

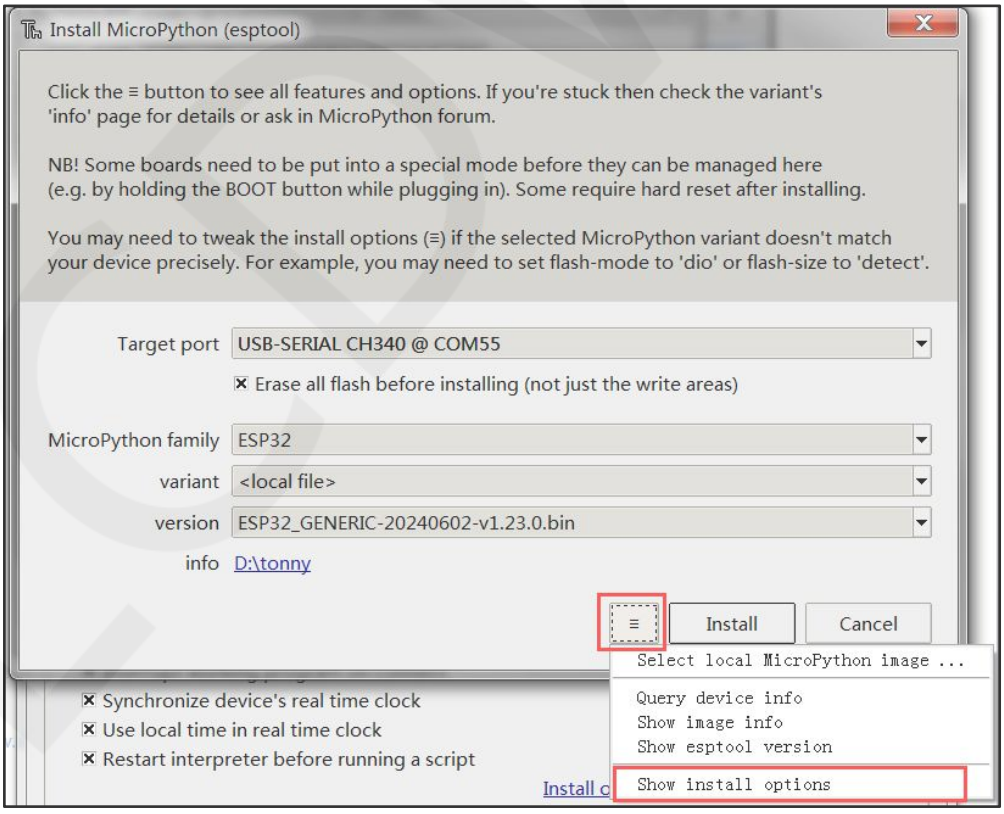

Figure 4.12 Open burn configuration information

After the configuration option is displayed, you can set the corresponding configuration option. If you are not clear about the configuration of the ESP32 module, keep the default configuration. The configuration interface is shown in the following figure:

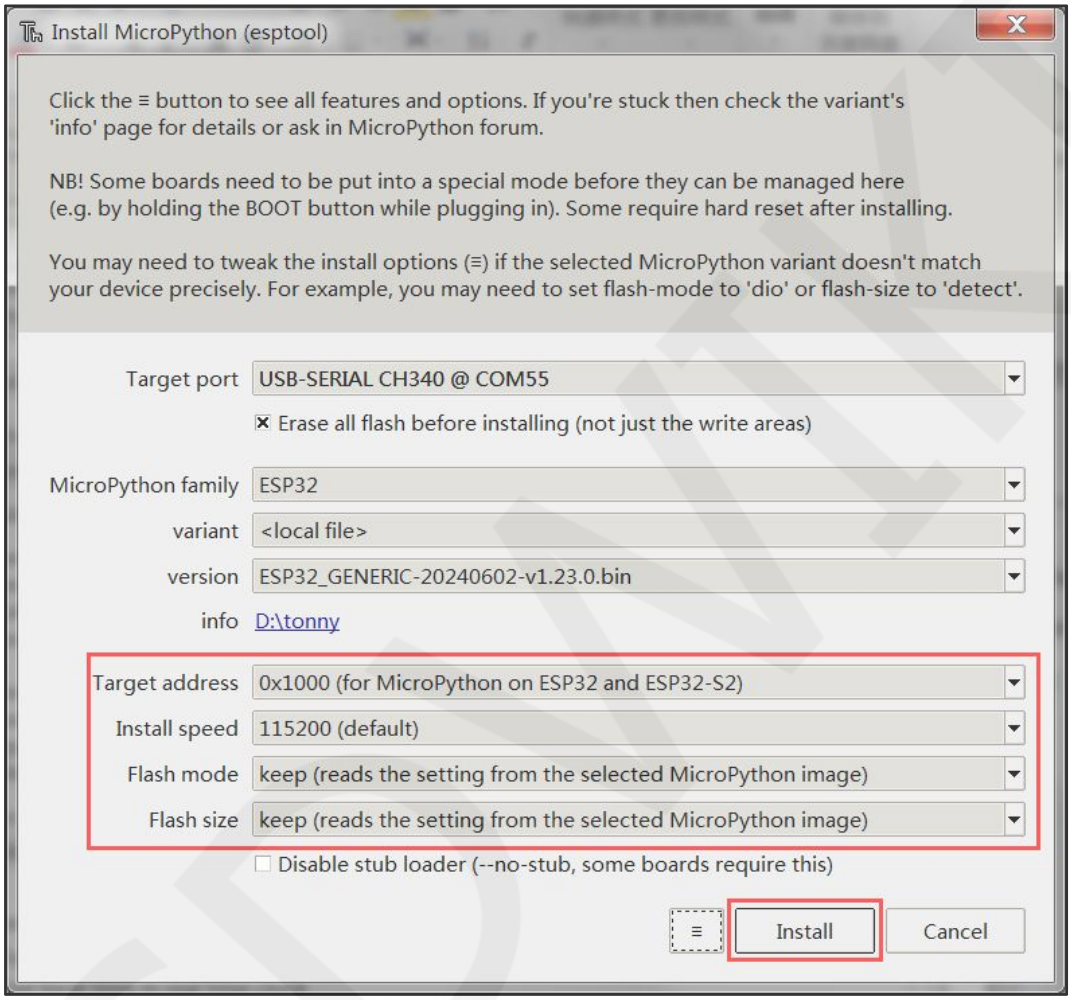

Figure 4.13 Configure the ESP32 module

Here are the configuration options:

Target sddress: Firmware burn address, there are two options: 0x0 and 0x1000, here using ESP32, so can only choose 0x1000

Install speed: Firmware burning rate, optional parameters are: 460800, 230400, 115200, 38400, 9600. Select the maximum rate supported by the USB-to-serial port on the ESP32 module.

Flash mode: Flash communication mode mounted on the ESP32. The options are Keep, DIO, QIO, DOUT, and QOUT. Where keep reads the configuration from the burned firmware; Both DIO and DOUT are

dual-wire SPI modes. DIO uses two data lines for communication in both the address and data stages, while DOUT only uses two data lines for communication in the data stage. Both QIO and QOUT are four-wire SPI modes. DIO uses four data lines for communication in both the address and data stages, while DOUT only uses four data lines for communication in the data stages. Select the connection mode based on the actual Flash connection mode of the ESP32 module.

Flash size: The value can be keep, detect, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB, 16MB, or 32MB. keep reads the configuration from the burned firmware, and detect obtains the capacity according to the Flash ID. Based on the actual capacity of the Flash.

Retain the default values for other configuration items.

After setting the configuration options, click the "**Install**" button, then enter the burning stage, you can see the burning progress, as shown in the picture below:

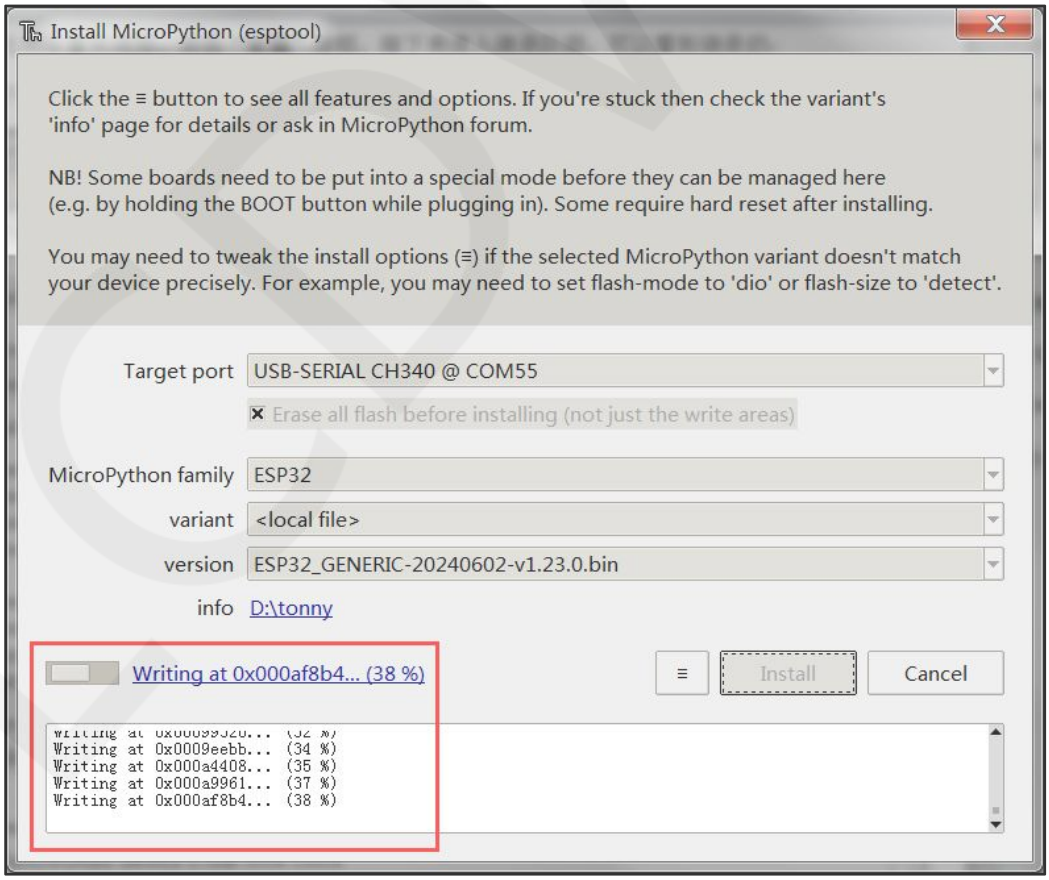

Figure 4.14 Burn ESP32 MicroPython firmware

When the words "**Done!** After the prompt, it indicates that burning has been completed, click the "**Close**" button, as shown below:

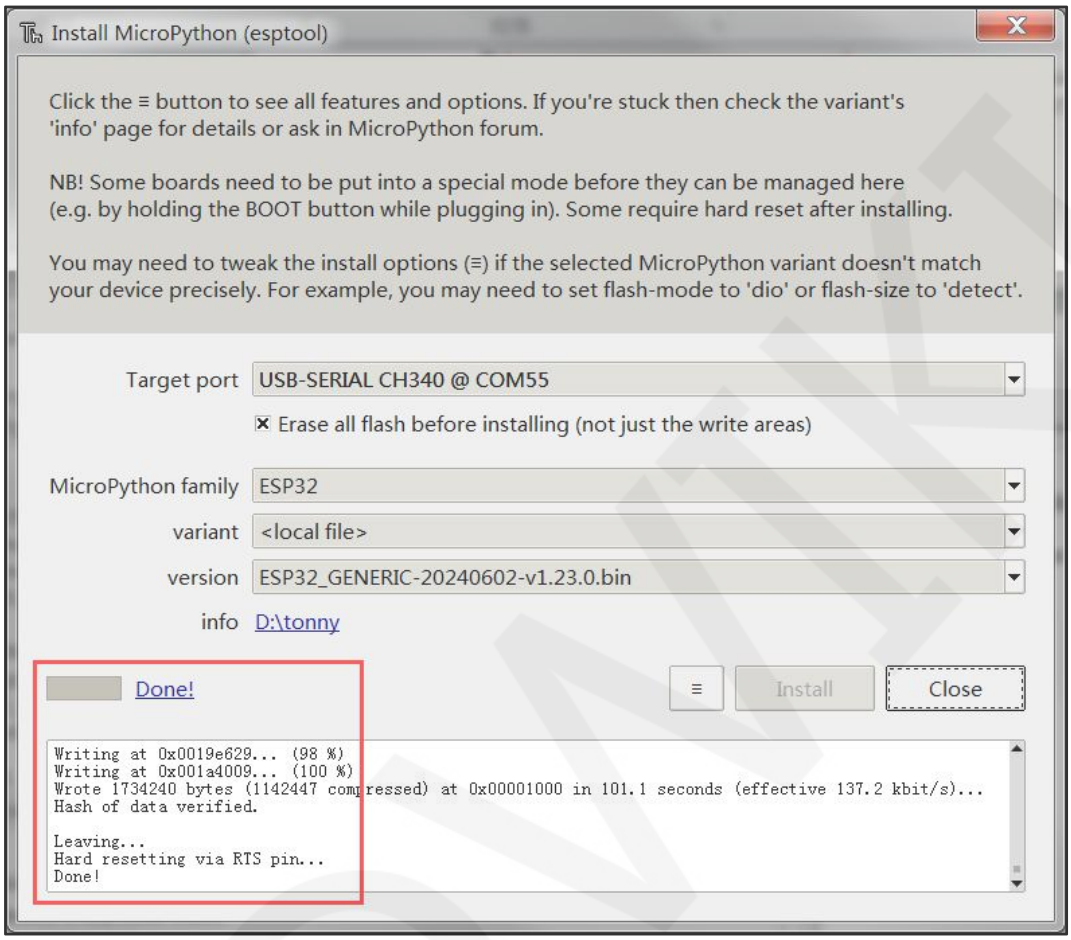

Figure 4.15 ESP32 MicroPython firmware has been burned

B. Online burning

Online burning refers to downloading the latest ESP32 MicroPython firmware from the MicroPython official website to the local computer through the network, and then burning. This method requires a computer to connect to the Internet.

After entering the burning interface, select the correct port first, and then configure **MicroPython family**, **variant**, **version** and other options. Generally, only configure the first two options, and version will be automatically obtained.

For more detailed configuration, refer to the description in the local burning mode.

After the configuration is complete, click the "**Install**" button, as shown in the picture below:

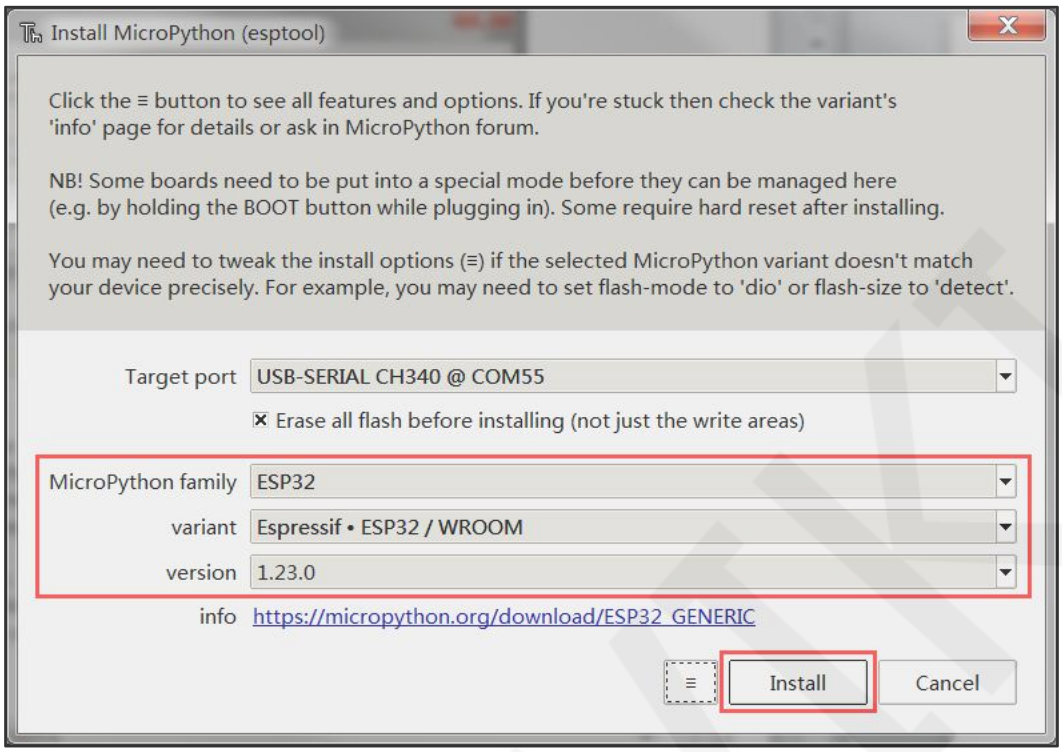

Figure 4.16 ESP32 MicroPython firmware online burning configuration

In the burning stage, you can see the burning progress, as shown in the figure below:

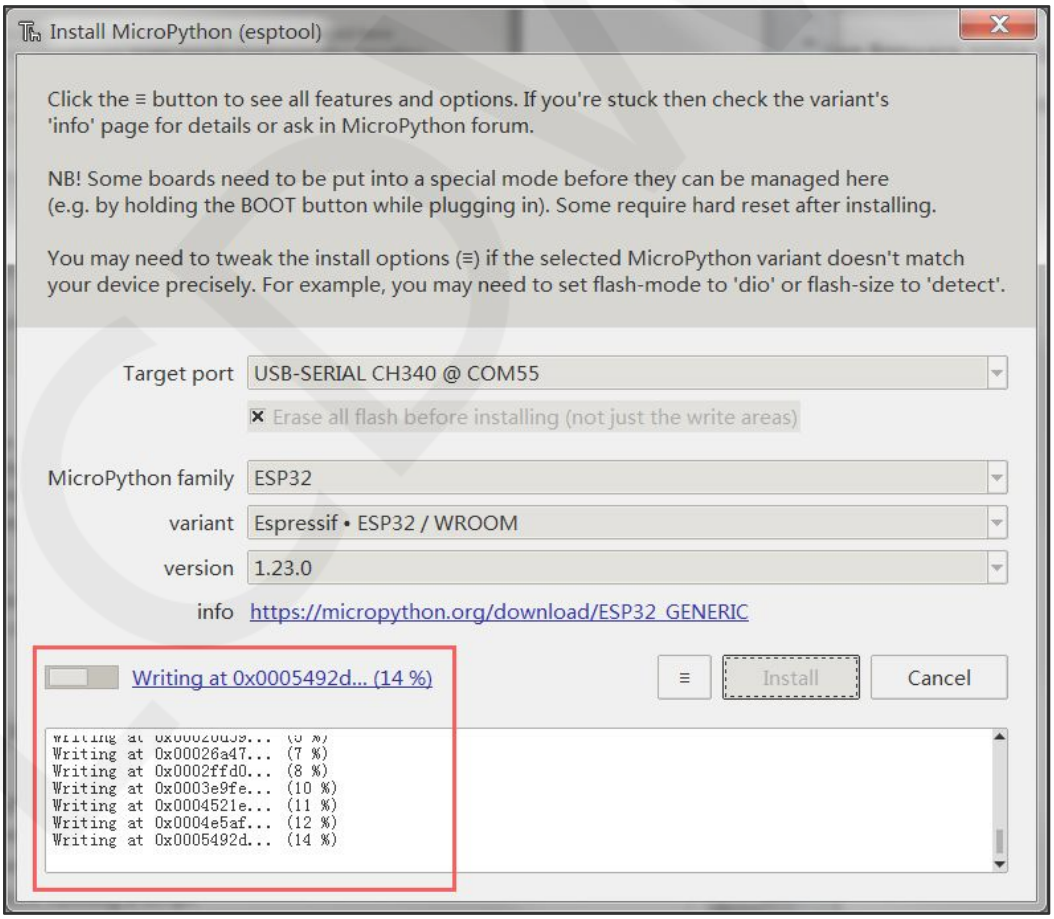

Figure 4.17 ESP32 MicroPython firmware is burned online

When the words "**Done!**" After the prompt, it indicates that burning has been completed, click the "**Close**" button, as shown below:

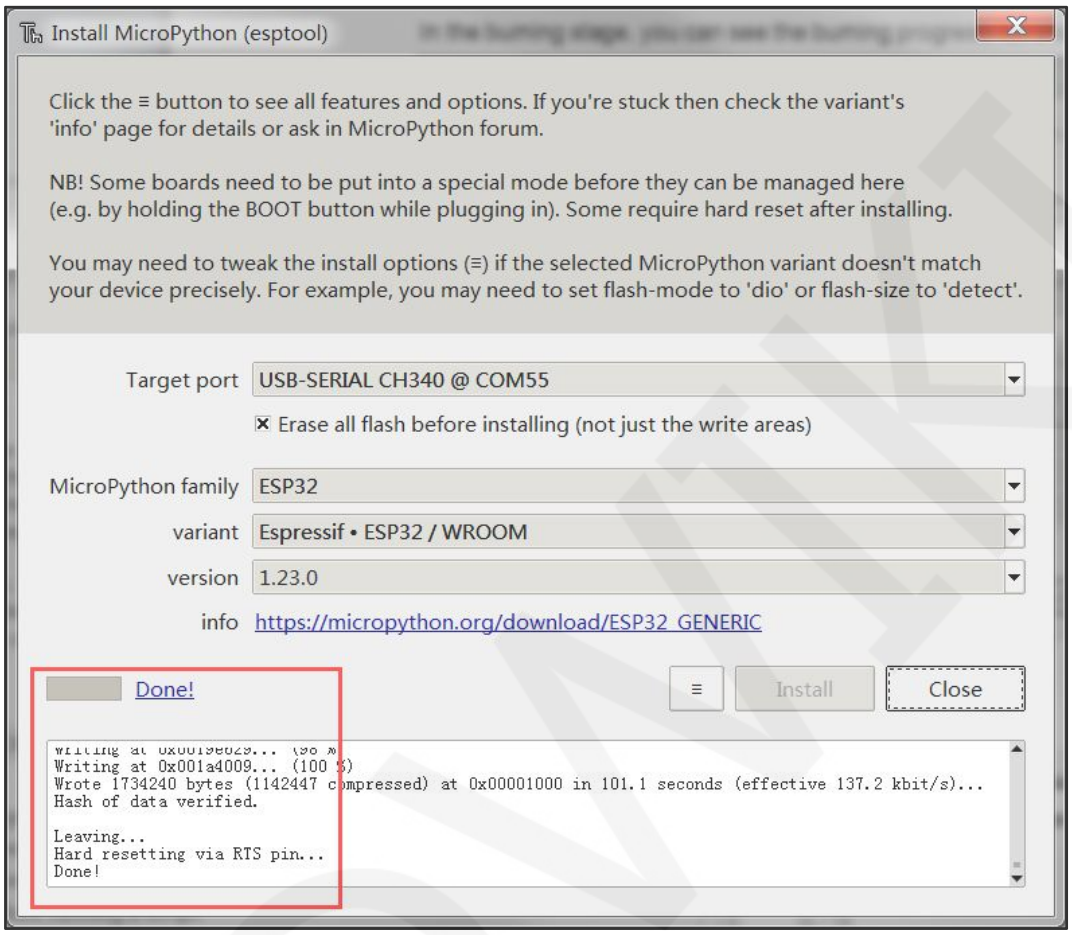

Figure 4.18 The ESP32 MicroPython firmware has been burned online

4.3.2. Burn firmware using flash_download_tool

The flash_download_tool tool can be downloaded from the official website.

Official download Website:

https://www.espressif.com.cn/support/download/other-tools

Figure 4.19 Download flash_download_tool from the official website

If it is not convenient to download it from the Internet, you can also obtain it from the "**7-工具软件_Tool_software**" folder in the document package. In the folder, find the **Flash_Download.zip** package and decompress it.

Open the flash_download_tool folder, find the exe file, double-click to open it, configure as shown below, and click the "**OK**" button.

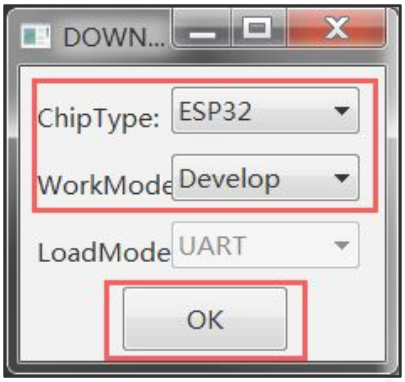

Figure 4.20 Configure the flash_download_tool tool

Perform the following steps to set flash_download_tool:

A. select the local computer to save the burning firmware, set the burning address, ESP32 burning address must be set to 0x1000,Otherwise, the ESP32 will not work properly after burning.

B. Set the SPI mode of SPI Flash, which is set as DIO according to the actual situation.

C. Set the SPI speed of SPI Flash. Set it as 80MHz according to the actual situation.

D. Set the COM port and the transmission rate of the COM port based on the actual situation.

E. Click the "**START**" button to start burning.

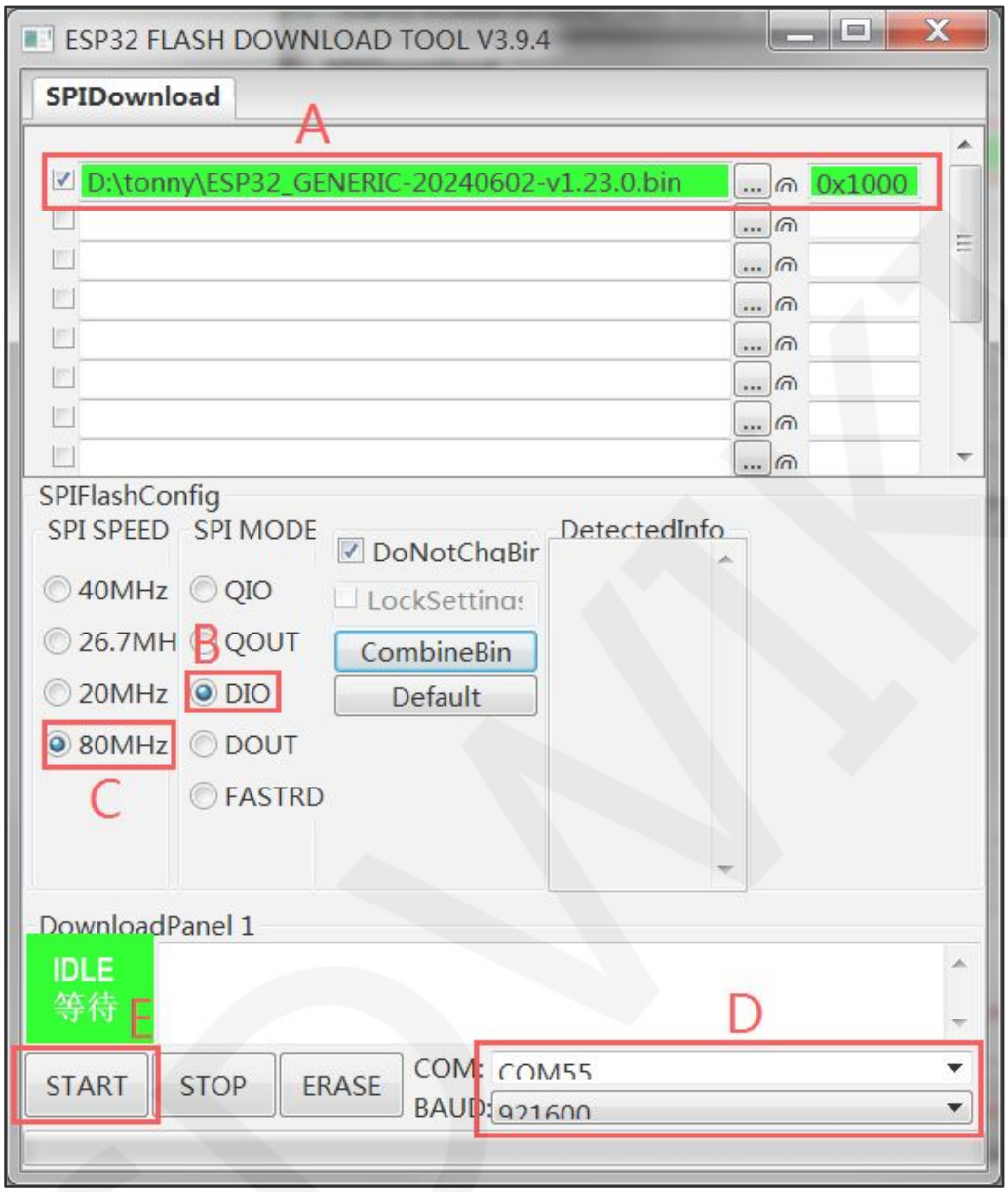

Figure 4.21 Use the flash_download_tool to burn the firmware

After starting the burning, you can see the burning prompt and progress bar changes. When the progress bar is finished and a "completed" prompt appears, it means that the burning is completed, as shown in the figure below:

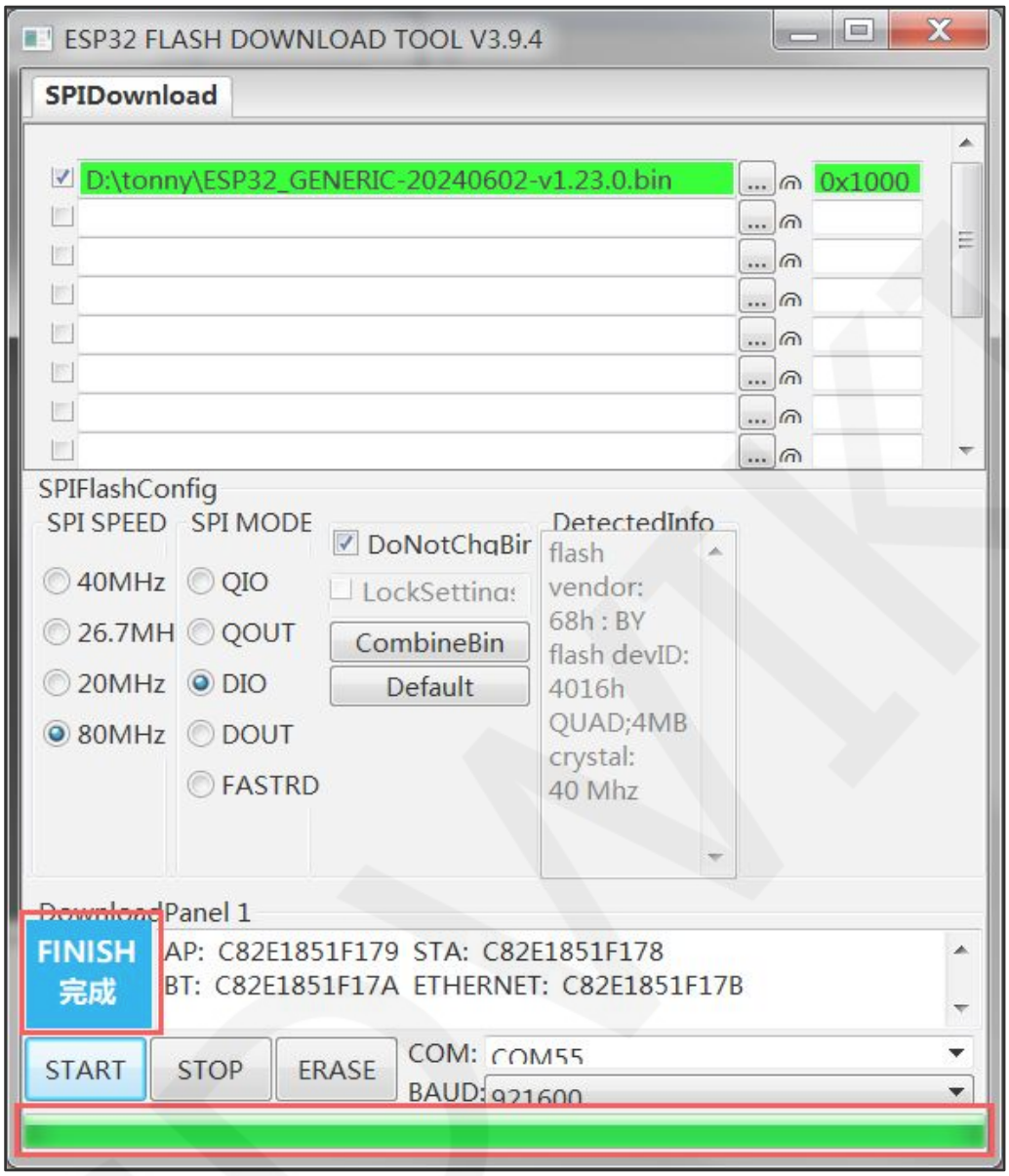

Figure 4.22 flash_download_tool The firmware is burned

5. Edit, Save and Run ESP32 MicroPython program

After the ESP32's MicroPython firmware is successfully burned, the next step is to develop MicroPython programs. It is preferred to connect the ESP32 module to the USB port of the computer to power on.

5.1. Configure the MicroPython interpreter

Click the software menu bar to **Run** -> **Configure Interpreter…**, or click the menu bar **Tools** -> **Options…** -> **Interpreter**, or click the button at the bottom of the Thonny software interface and select Configure Interpreter. On the interpreter interface, select the **MicroPython(ESP32)** interpreter, select the actual port number, and then click the "**OK**" button to save and exit. As shown in the picture below:

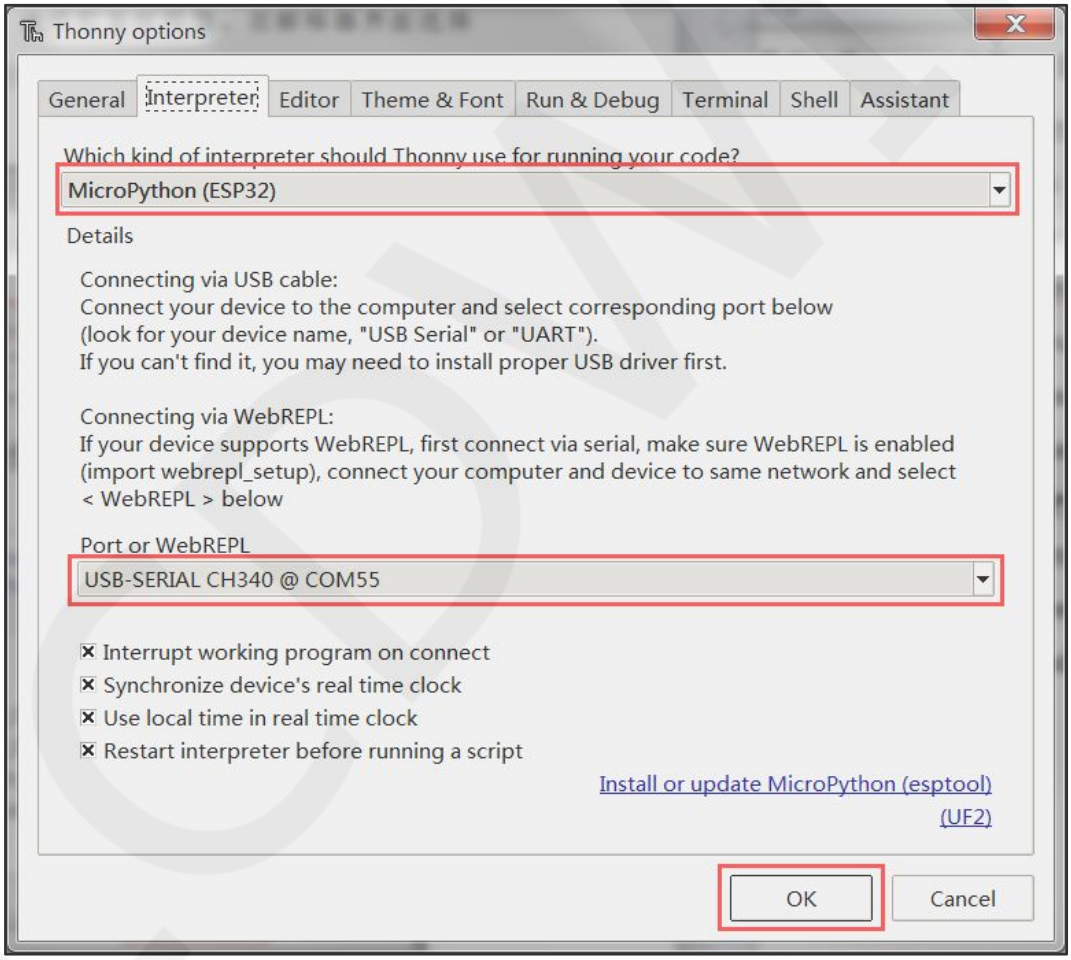

Figure 5.1 Configure the MicroPython interpreter

5.2. Edit ESP32 MicroPython programs

What is introduced here is to create a program file and edit, if the program file already exists, then you can directly open the existing file for editing and use.

Click **File** -> **New** button, or click the icon on the toolbar, or press the shortcut key "**Ctrl+N**" to create a new program file, as shown below:

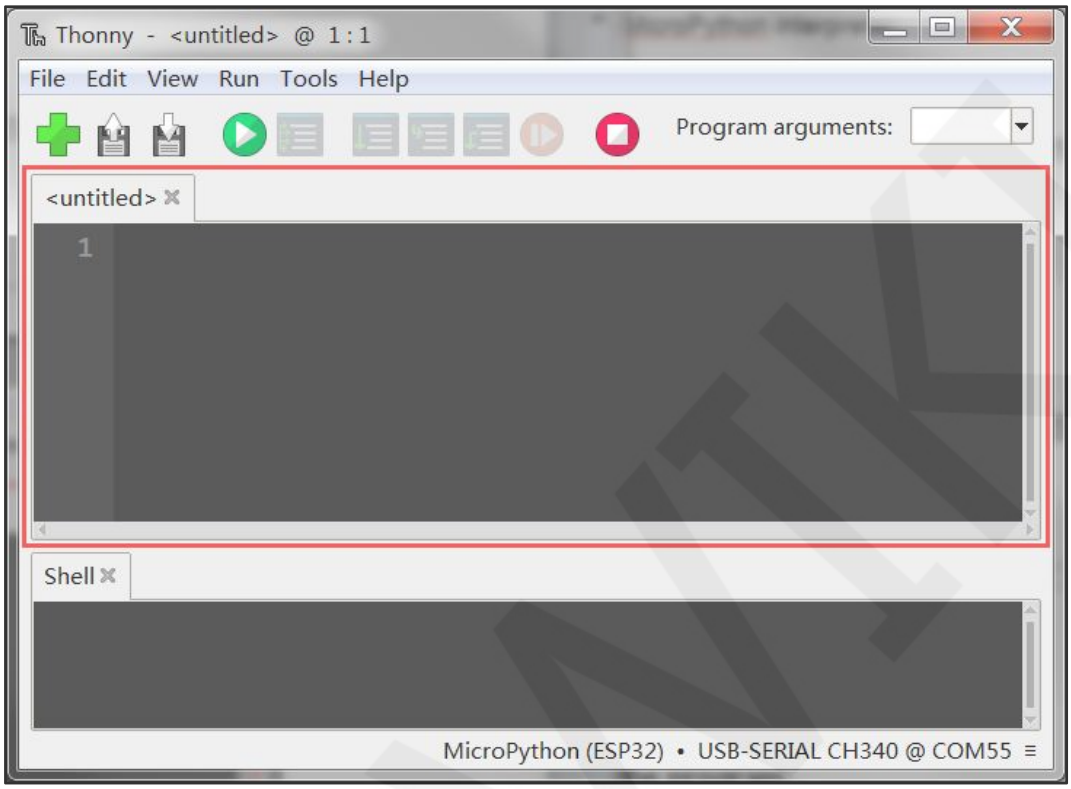

Figure 5.2 New program file

Enter the following in the program editing window:

```
import machine
import esp

esp32_id    = machine.unique_id()

id_str = ''.join(['{:02x}'.format(byte) for byte in esp32_id])

f_size = esp.flash_size()

print("Unique identifier for ESP32:",id_str)
print("ESP32 Flash Size:{}B".format(f_size))
```

Figure 5.3 Input program content

Enter the program's editing window, as shown below:

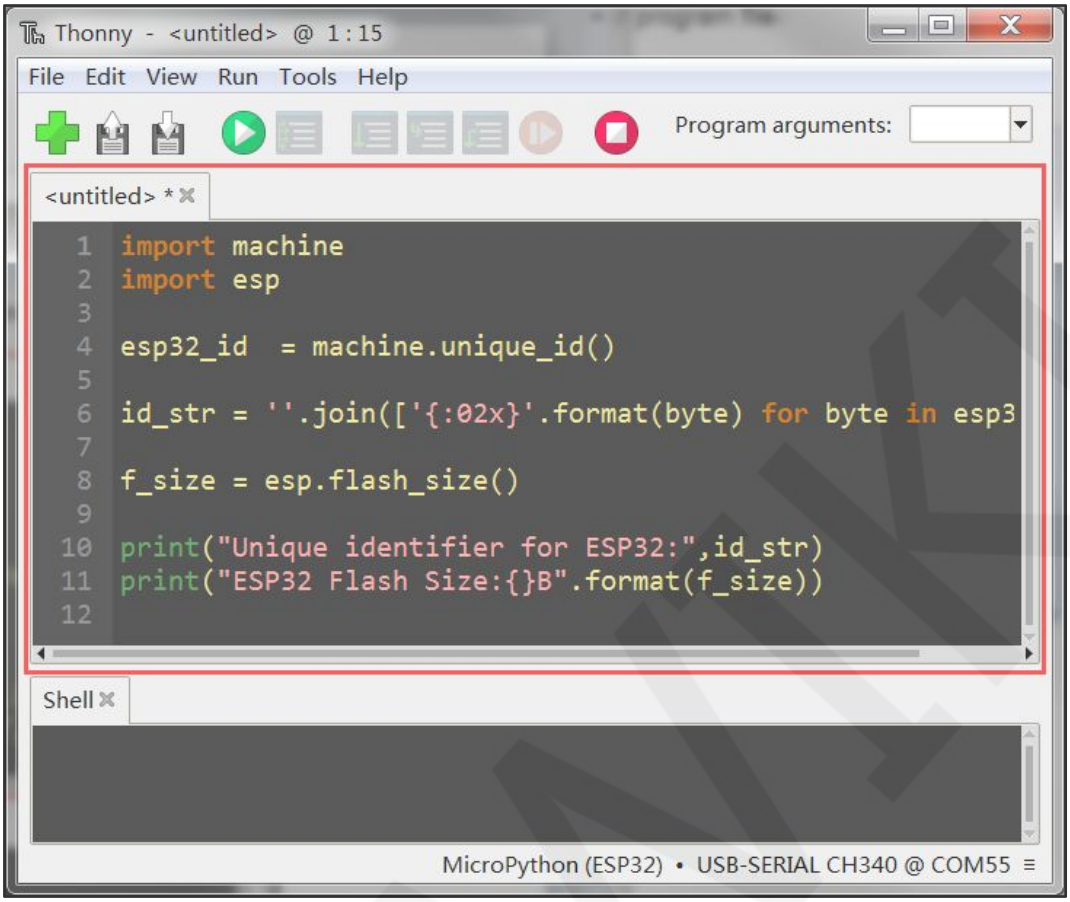

Figure 5.4 Program edit window content

5.3. Save and run the ESP32 MicroPython program

After editing the program, you need to run the program to check for errors and exceptions. If **Tools -> Options... -> Run & Debug** menu under the "**Allow running unnamed programs**" option is selected, you can directly run the edited program, temporary check, improve development efficiency, otherwise need to save before running. As shown in the picture below:

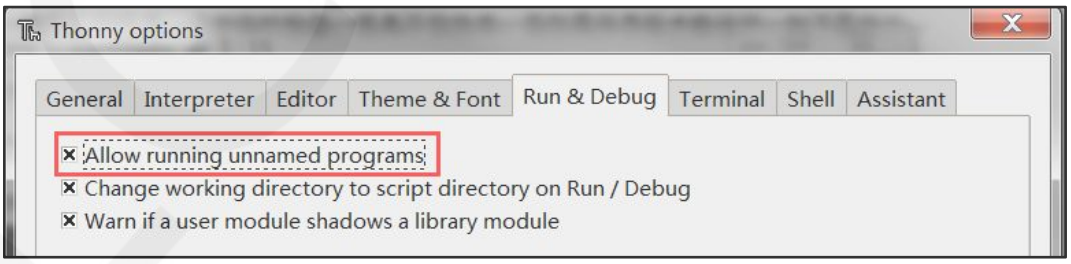

Figure 5.5 Program run configuration

Note: If the Run button in the toolbar is gray (can't be clicked), click the stop/restart button to start the back-end process.

After the ESP32 MicroPython program is edited, it needs to be saved, either on the local computer or in the ESP32 module.

A. Save it on the local computer

If Thonny is not connected to the ESP32 module, it can only be saved on the local computer. Click **File** -> **Save**, or click the toolbar **Save** button  , or press the "**Ctrl+S**" shortcut key, then enter the file name, and then select the destination folder to save. If Thonny is connected to ESP32 module, after the above operation, the save location selection interface will pop up, select "**This computer**", as shown in the following picture:

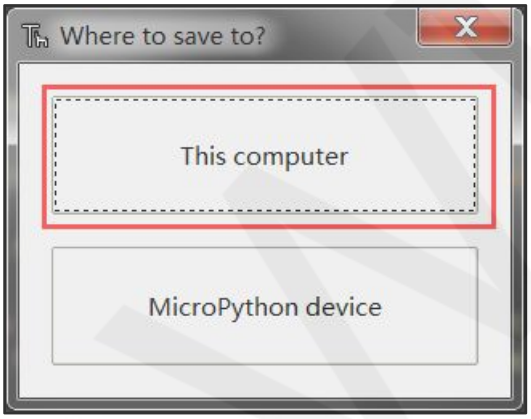

Figure 5.6 Select save location 1

B.Save in ESP32 module

The ESP32 module must be connected to Thonny software. There are two types of saving: one is to create a file and save it, the other is to upload it from the local computer.

To save a new file, click **File** -> **Save**, or click the **Save** button  in the toolbar, or press the **Ctrl+S** shortcut key, select "**MicroPythons device**" in the pop-up save location screen, and then enter the file name, click "**OK**" button to save. As shown in the picture below:

Figure 5.7 Select save location 2

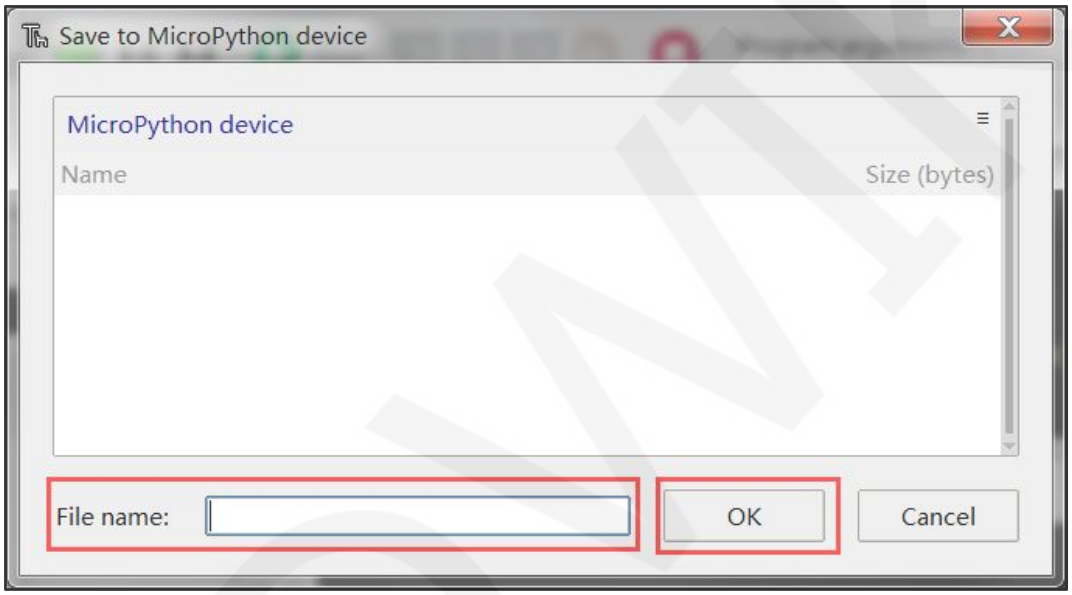

Figure 5.8 Save the new file to the MicroPython device

When uploading from the local computer, click **View** -> **Files** in the menu bar to open the local file browsing window, and then find the file to be uploaded, select the file and click the right mouse button, and choose "**Upload to /**" option in the pop-up menu. At this time, enter the upload file stage.

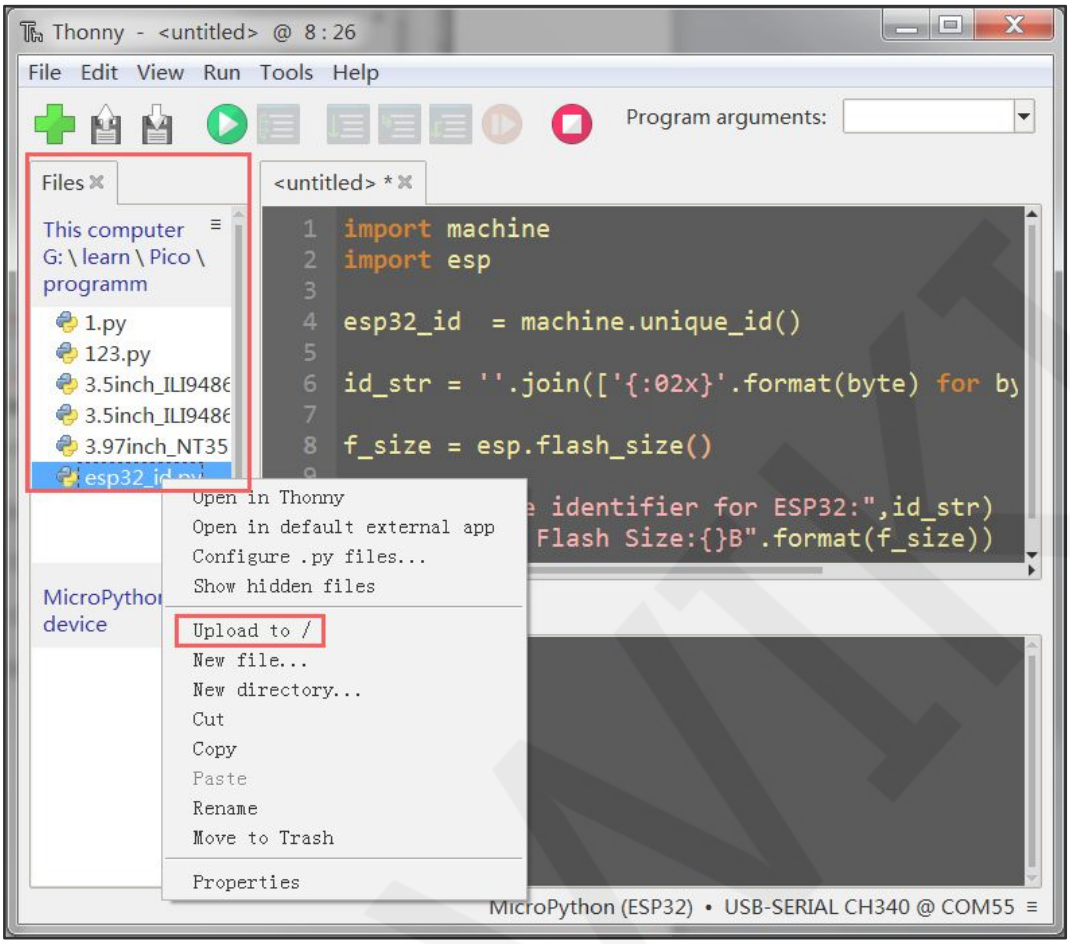

Figure 5.9 Save the local file to the MicroPython device

After uploading, you can see the file name in the MicroPython device bar, as shown below:

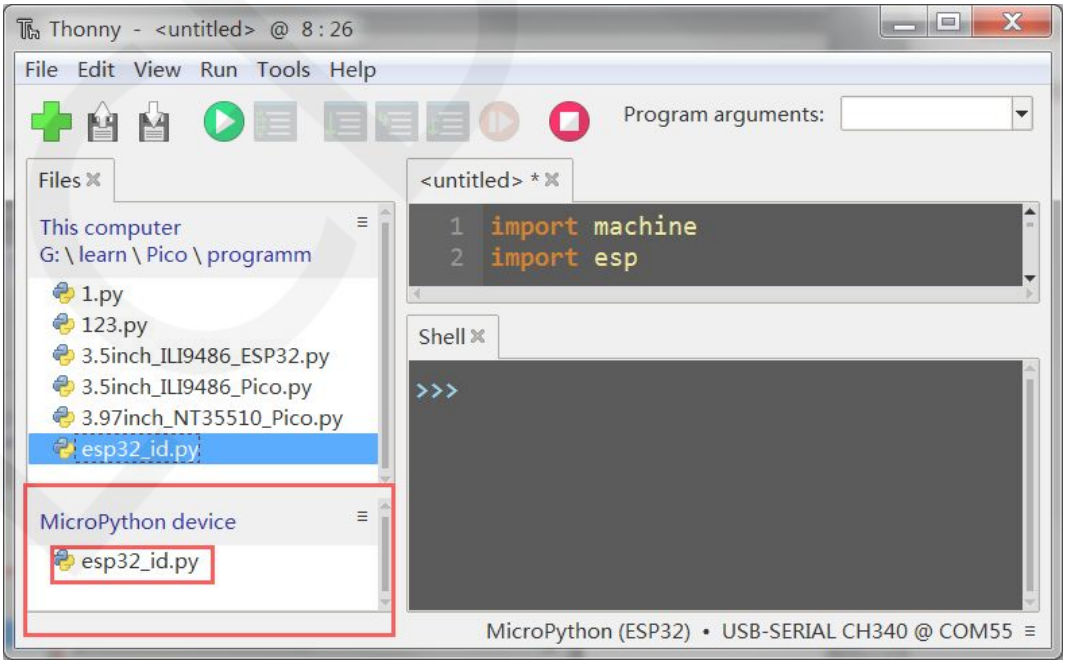

Figure 5.10 The local file has been saved to the MicroPython device

Once the file is saved, run the file next. First of all, open the file (open the local computer file or the file in the MicroPython device can be), and then click **Run** -> **Run current script**, or click the **Run** button  in the toolbar, or press the "**F5**" shortcut key, run the program file, you can see the running result in the Shell window, as shown in the following figure:

Figure 5.11 Run program file

Note: When the above operation runs the program file, it is a temporary debugging run, which needs to use Thonny software. If you want to power on the ESP32 device to run the program directly, you need to rename the object program file to "**main.py**" and save it to the ESP32 device.