Activity 1	**Installing ESP32 Board in the Arduino IDE**	**31/1/2025**

There are several development platforms available for programming the ESP32. You can go with:

- Arduino IDE – intended for those who are familiar with Arduino
- Espruino – JavaScript SDK and firmware closely emulating Node.js
- Mongoose OS – An operating system for IoT devices that is recommended by Espressif Systems and Google Cloud IoT
- MicroPython – Implementation of Python 3 for microcontrollers
- SDK provided by Espressif – Official SDK to take advantage of all ESP32 features

When compared to other platforms, the Arduino IDE is the most user-friendly for beginners. While it may not be the ideal platform for working with the ESP32, it is a program that most

Step 1: Installing or Updating the Arduino IDE

The first step in installing the ESP32 Arduino core is to have the latest version of the Arduino IDE installed on your computer. If you haven't already, we recommend that you do so right away. Search for:

Latest Arduino IDE

Step 2: Installing the USB-to-Serial Bridge Driver

There are numerous ESP32-based development boards available. Depending on the design, you may need to install additional drivers for your USB-to-serial converter before you are able to upload code to your ESP32.

For example, the ESP32 DevKit V1 uses the CP2102 to convert USB signals to UART signals, whereas the WeMos ESP32 Lite uses the CH340G. The ESP32-CAM, on the other hand, lacks an onboard USB-to-serial converter and requires a separate module.

Make sure to inspect your board carefully to identify the USB-to-serial converter that is present. You'll probably have either CP2102 or CH340 populated on the board.

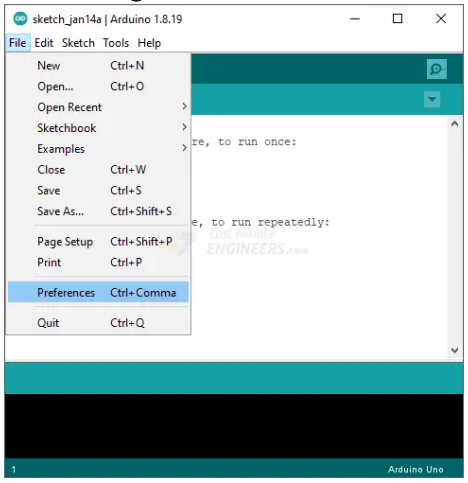
If you've never installed drivers for these USB-to-serial converters on your computer before, you should do so right now. Search for:

CP210x USB to UART Bridge VCP Drivers

CH340 drivers

Step 3: Installing the ESP32 Arduino Core

Launch the Arduino IDE and navigate to File > Preferences.

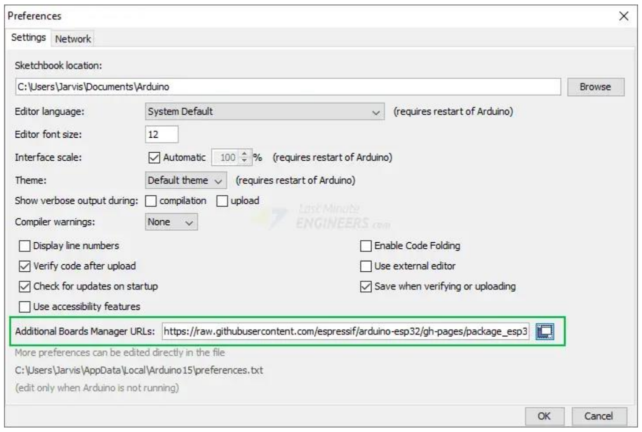


```
re, to run once:

e, to run repeatedly:
```

Fill in the "Additional Board Manager URLs" field with the following.

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

Then, click the "OK" button.



Sketchbook location: C:\Users\Jarvis\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☐ Display line numbers ☐ Enable Code Folding

☑ Verify code after upload ☐ Use external editor

☑ Check for updates on startup ☑ Save when verifying or uploading

☐ Use accessibility features

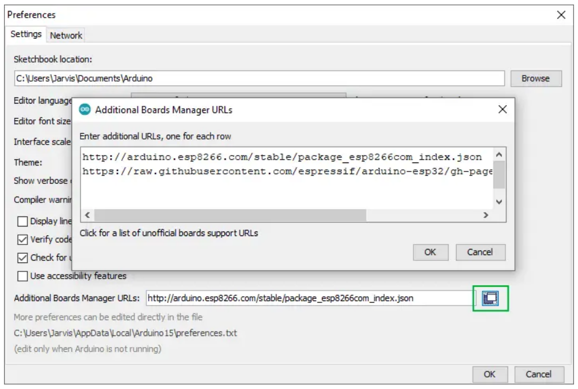
Additional Boards Manager URLs: https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp3

More preferences can be edited directly in the file

C:\Users\Jarvis\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

If you have already entered the URL for the ESP8266 boards or any other board, you can click on the icon to the right of the field to open a window where you can add additional URLs, one for each row.



Sketchbook location:
C:\Users\Jarvis\Documents\Arduino

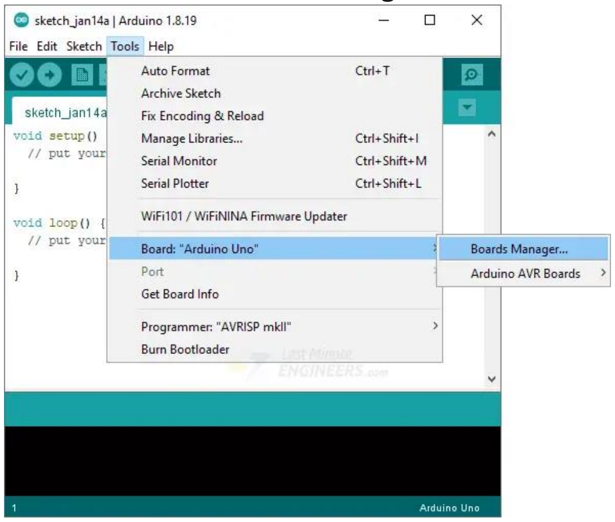
Enter additional URLs, one for each row

```
http://arduino.esp8266.com/stable/package_esp8266com_index.json
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-page
```

Additional Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

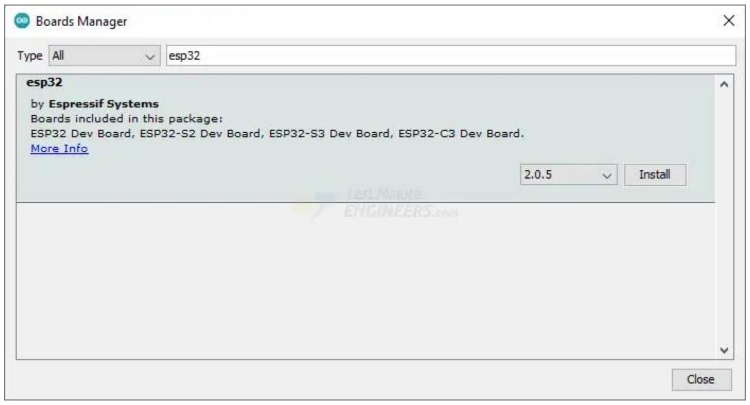
More preferences can be edited directly in the file
C:\Users\Jarvis\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

Now navigate to Tools > Board > Boards Manager…



```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

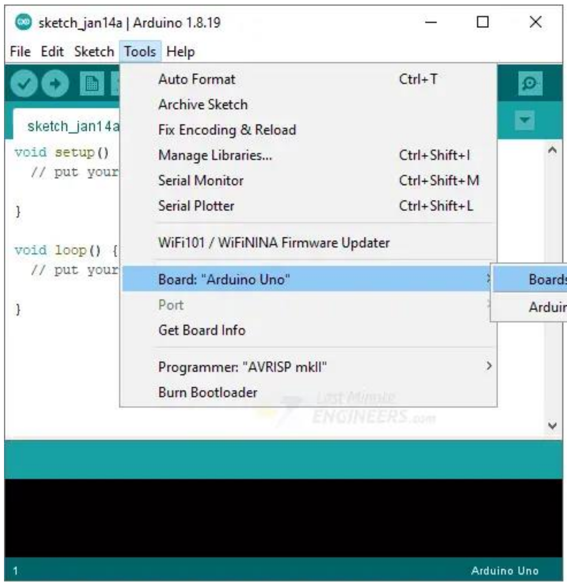
Filter your search by entering ‘esp32‘. Look for ESP32 by Espressif Systems. Click on that entry, and then choose Install.



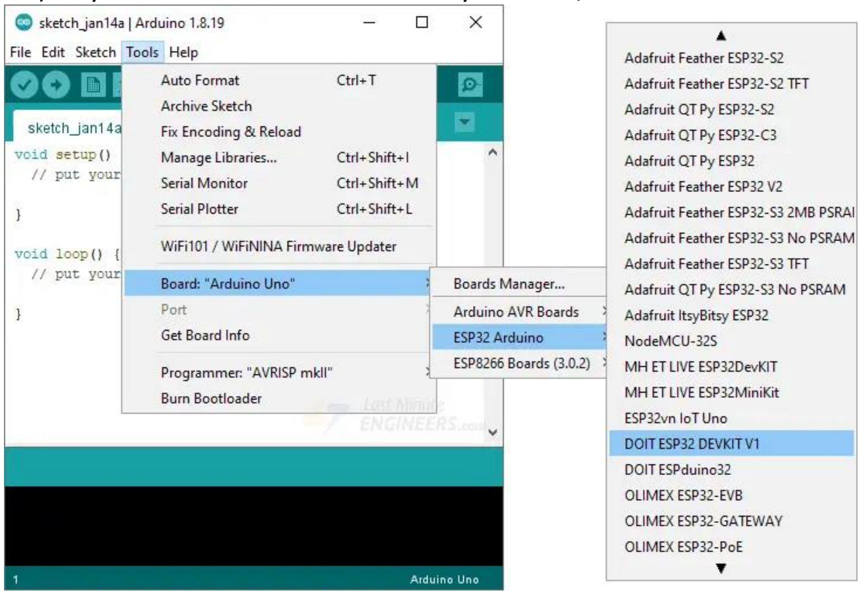
esp32
by **Espressif Systems**
Boards included in this package:
ESP32 Dev Board, ESP32-S2 Dev Board, ESP32-S3 Dev Board, ESP32-C3 Dev Board.

Step 4: Selecting the Board and Port

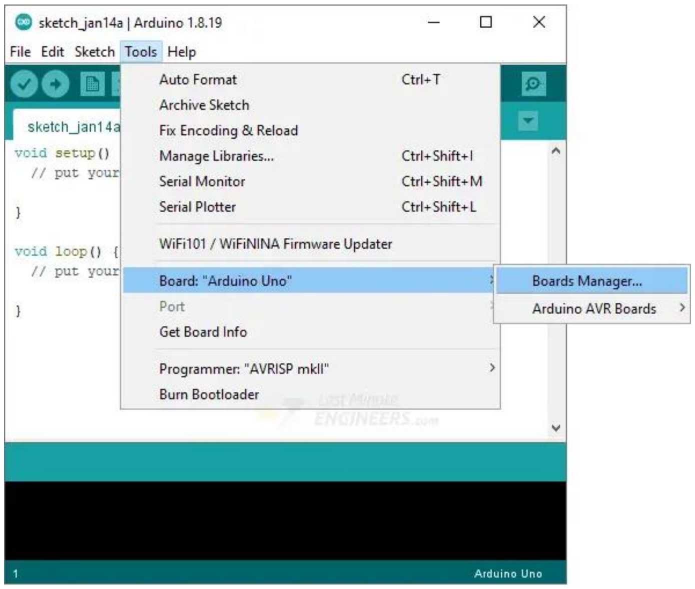
After installing the ESP32 Arduino Core, restart your Arduino IDE and navigate to Tools > Board to ensure you have ESP32 boards available.


sketch_jan14a | Arduino 1.8.19
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Uno

Now select your board in the Tools > Board menu (in our case, it's the DOIT ESP32 DEVKIT V1). If you are unsure which board you have, select ESP32 Dev Module.



Finally, connect the ESP32 board to your computer and select the Port.


sketch_jan14a | Arduino 1.8.19
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Uno

That’s it! You can now begin writing code for your ESP32 in the Arduino IDE.

Step 5: Testing the Installation

Once you’ve finished the preceding steps, you are ready to test your first program with your ESP32! Launch the Arduino IDE. If you disconnected your board, plug it back in.

Let’s upload the most basic sketch of all – Blink!

This sketch uses the on-board LED that most ESP32 development boards have. This LED is connected to digital pin D2, and its number may vary from board to board.

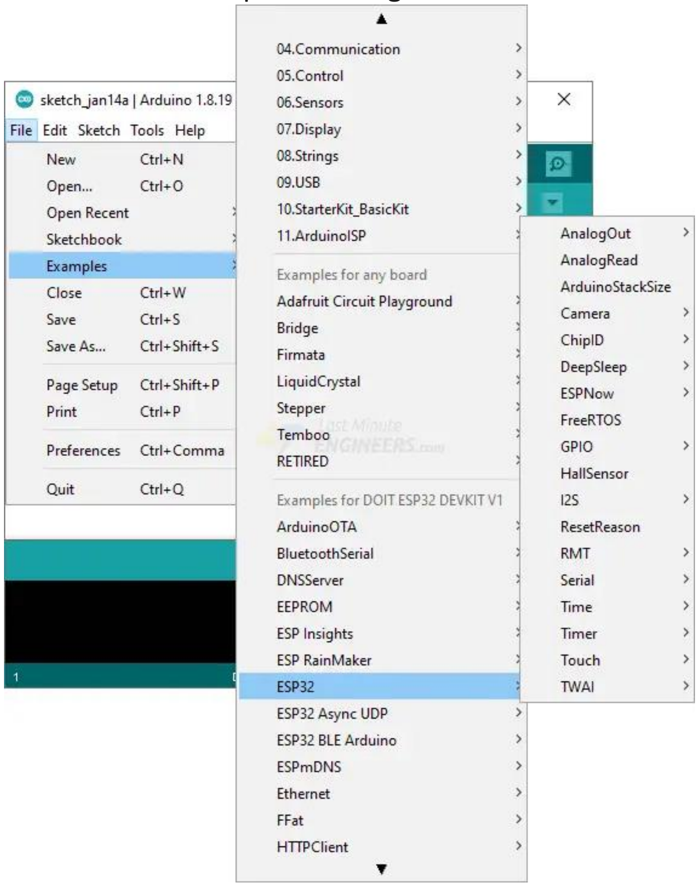
```
int ledPin = 2;

void setup() {
		pinMode(ledPin, OUTPUT);
}

void loop() {
		digitalWrite(ledPin, HIGH);
		delay(500);
		digitalWrite(ledPin, LOW);
		delay(500);
}
```

If everything worked, the on-board LED on your ESP32 should now be blinking! To execute the sketch, you may need to press the EN button on your ESP32.

Congratulations! You have just programmed your first ESP32!

More ESP32 Examples

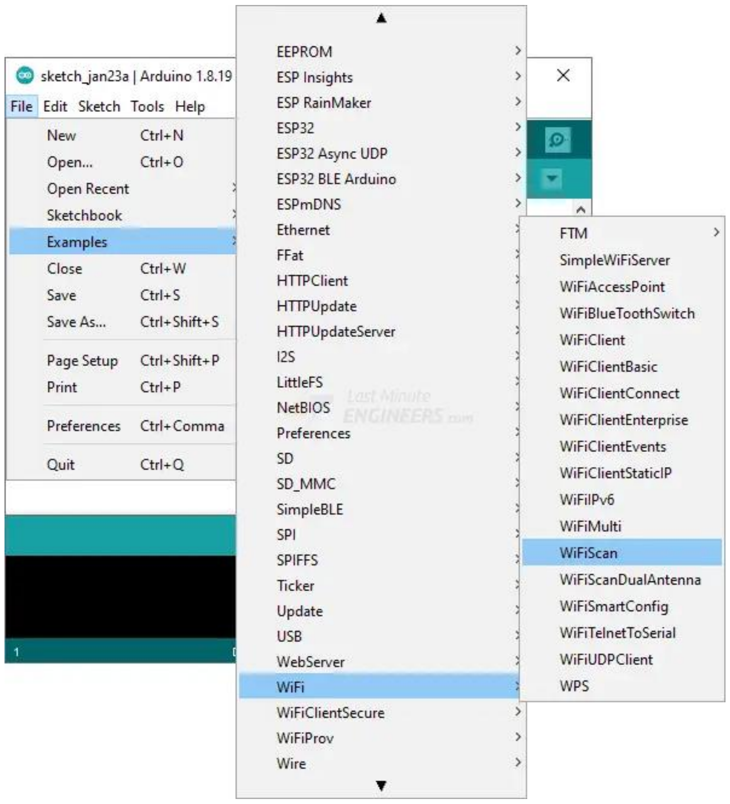
The ESP32 Arduino core includes several examples that demonstrate everything from scanning for nearby networks to building a web server. To access the example sketches, navigate to File > Examples > ESP32.
You will see a selection of example sketches. You can choose any of them to load the sketch into your IDE and start experimenting with it.



ESP32 Example: WiFi Scan

Let’s try to run one of the ESP32 example sketches, which demonstrates how to use the WiFi library to scan nearby WiFi networks and print the results.

You can find this example under File > Examples > WiFi > WiFiScan.



Load the WiFiScan sketch from the example sketches into your Arduino IDE.

```
#include "WiFi.h"

void setup()
{
  Serial.begin(115200);

  // Set WiFi to station mode and disconnect from an AP if it was previously connected.
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(100);

  Serial.println("Setup done");
}

void loop()
{
  Serial.println("Scan start");

  // WiFi.scanNetworks will return the number of networks found.
  int n = WiFi.scanNetworks();
  Serial.println("Scan done");
  if (n == 0) {
    Serial.println("no networks found");
  } else {
    Serial.print(n);
    Serial.println(" networks found");
    Serial.println("Nr | SSID                             | RSSI | CH | Encryption");
    for (int i = 0; i < n; ++i) {
      // Print SSID and RSSI for each network found
      Serial.printf("%2d",i + 1);
```

```
        Serial.print(" | ");
        Serial.printf("%-32.32s", WiFi.SSID(i).c_str());
        Serial.print(" | ");
        Serial.printf("%4d", WiFi.RSSI(i));
        Serial.print(" | ");
        Serial.printf("%2d", WiFi.channel(i));
        Serial.print(" | ");
        switch (WiFi.encryptionType(i))
        {
        case WIFI_AUTH_OPEN:
          Serial.print("open");
          break;
        case WIFI_AUTH_WEP:
          Serial.print("WEP");
          break;
        case WIFI_AUTH_WPA_PSK:
          Serial.print("WPA");
          break;
        case WIFI_AUTH_WPA2_PSK:
          Serial.print("WPA2");
          break;
        case WIFI_AUTH_WPA_WPA2_PSK:
          Serial.print("WPA+WPA2");
          break;
        case WIFI_AUTH_WPA2_ENTERPRISE:
          Serial.print("WPA2-EAP");
          break;
        case WIFI_AUTH_WPA3_PSK:
          Serial.print("WPA3");
          break;
        case WIFI_AUTH_WPA2_WPA3_PSK:
          Serial.print("WPA2+WPA3");
          break;
        case WIFI_AUTH_WAPI_PSK:
          Serial.print("WAPI");
          break;
        default:
          Serial.print("unknown");
        }
        Serial.println();
        delay(10);
      }
    }
    Serial.println("");

    // Delete the scan result to free memory for code below.
    WiFi.scanDelete();

    // Wait a bit before scanning again.
    delay(5000);
}
```

Once you have uploaded the sketch, open the serial monitor at baud rate 115200 and press the EN button on the ESP32. You should see the SSID, RSSI, WiFi channel, and encryption for each discovered network.

Troubleshooting

Some ESP32 boards enter flashing mode automatically, and the code is uploaded successfully, while others do not, and you may receive the "A fatal error occurred: Failed to connect to ESP32… Timed out waiting for packet header…" error:

This is a common issue, and it indicates that your ESP32 is not in flashing or uploading mode. You can resolve this issue by following the steps outlined below.

- Hold down the BOOT button on your ESP32 board.
- Press the Upload button in the Arduino IDE to upload a new sketch.
- Release the BOOT button when the Writing at 0x00001000… (100%) message appears in the Arduino IDE log after Connecting….
- You should now see the "Done uploading" message.

That's it. Now, press the EN button to restart the ESP32 and run the newly uploaded sketch.

Remember! You'll have to repeat that button sequence every time you want to upload a new sketch.