

## Workshop III

# Arduino IDE Setup & Potentiometers

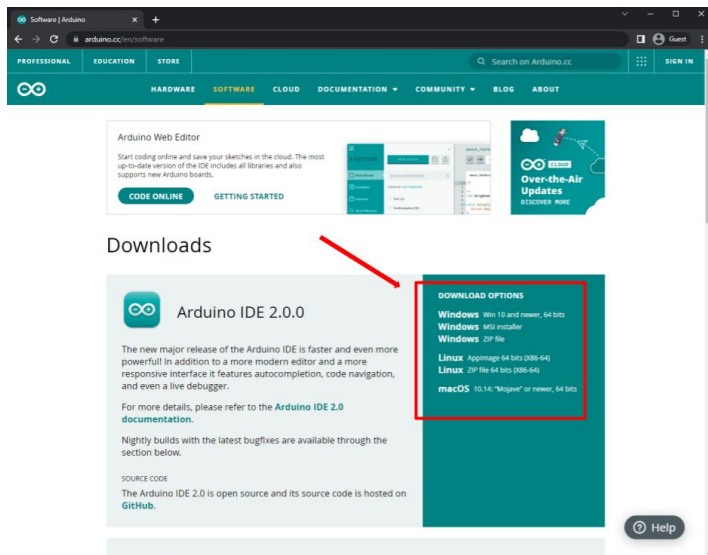
© 2024 Open Project Space, Institute of Electrical and Electronics Engineers at the University of California, Irvine. All Rights Reserved.

## **SECTION I**

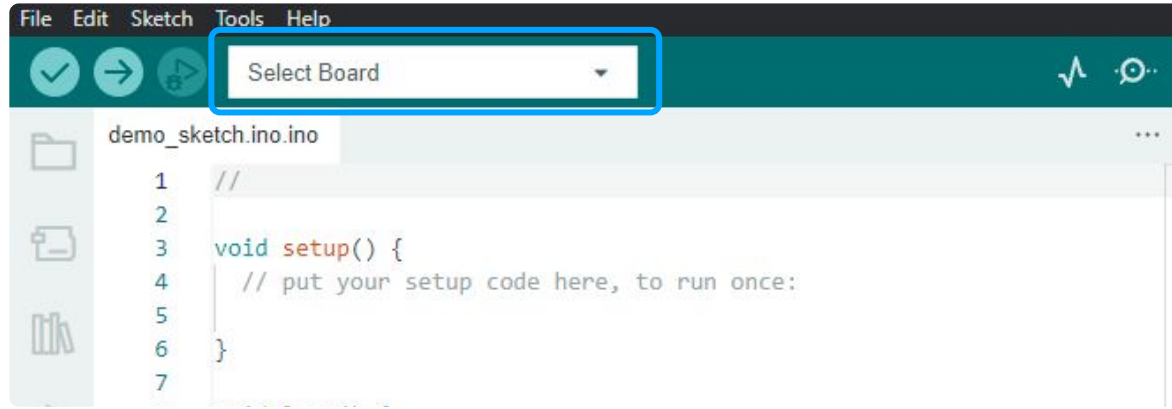
# **Review: ESP32 Board Setup**

# Arduino IDE Setup for ESP32

- If you haven't yet, visit the [ESP32 Guide](#) on the OPS Website for more information on how to setup Arduino IDE (Watch [YouTube video](#) for how-to)

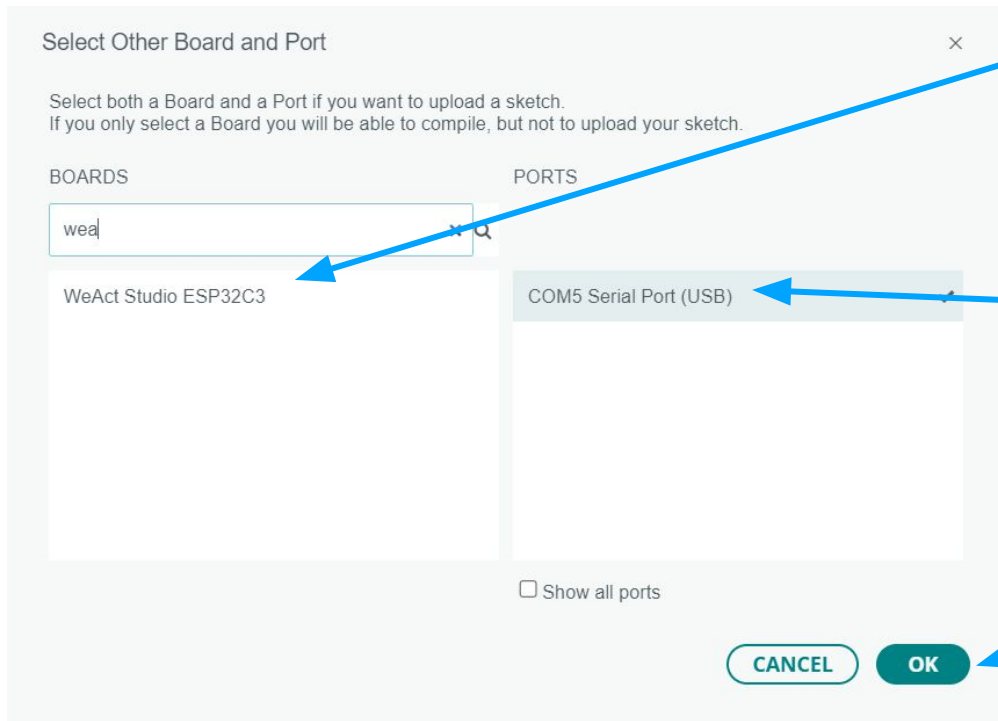


# Setting the Target Board



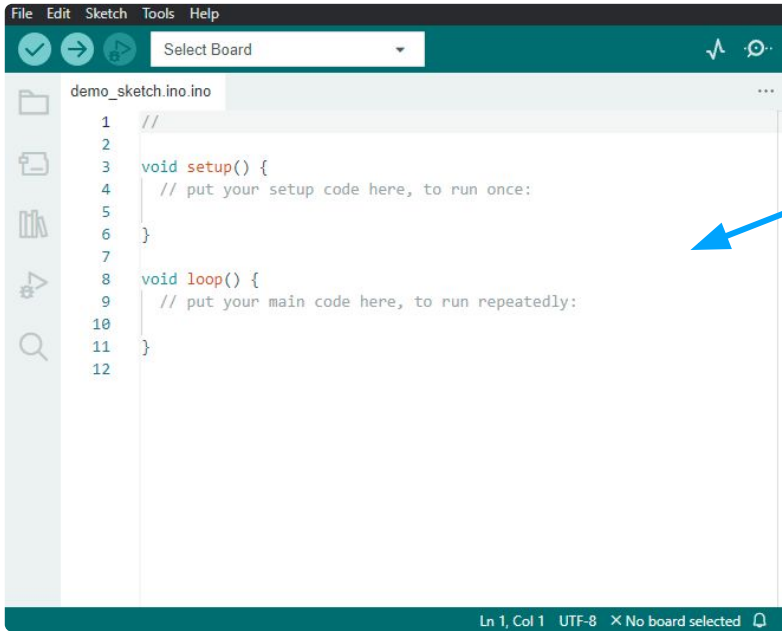
- Before we can upload a sketch, we must **connect the ESP32 board via the USB-C to USB-A cable to the computer** and configure the IDE to the correct board and USB port
- Open the **Select Board** dropdown and select an available ESP32 board
  - If no option appears, click **Select other board and port...**

# Setting the Target Board (Cont'd)



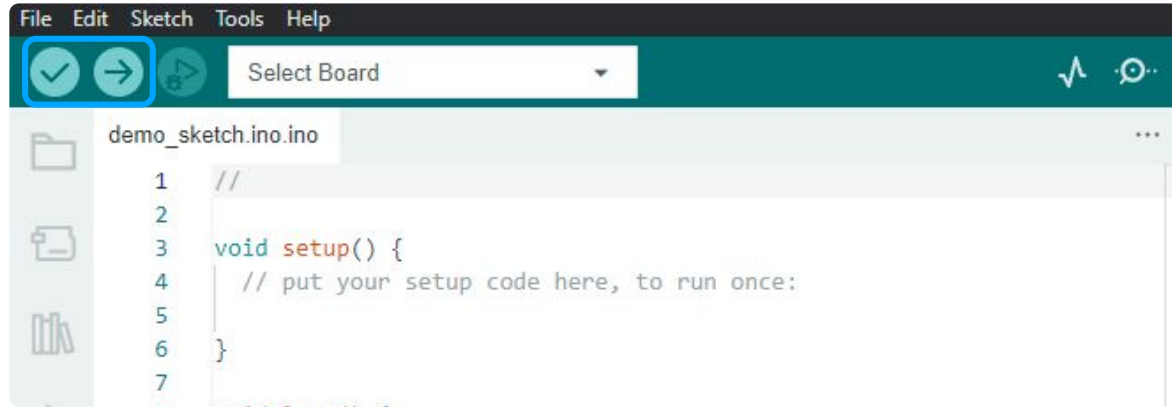
- In the popup window, search **Boards** and select **WeAct Studio ESP32C3**
- From the available items under **Port**, select the USB port to which the board is currently connected
- Confirm the selections by clicking **OK**

# Editing a Sketch



- Sketches can be modified from the **code editor**, which appears just below the menu
- New sketches opened in the editor come with **template code**

# Verifying and Uploading a Sketch



- Before uploading the sketch, select **Verify** (the **checkmark** icon) to compile the sketch
- Once the sketch compiles successfully, select **Upload** (the **right arrow** icon) to upload the compiled sketch to the Arduino board

## **SECTION II**

# **Review: ESP32 Boot Mode (For Programming)**



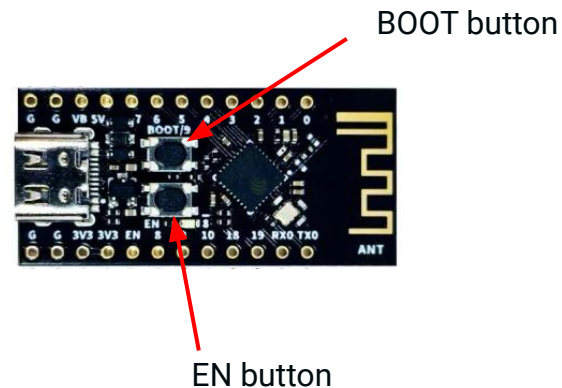
# ESP32 Boot Mode

- You will need to enter Boot Mode on the ESP32 whenever you upload code to the ESP32 through Arduino IDE

## How to Enter ESP32 Boot Mode:

1. Hold down the **BOOT button** for 2 secs
2. While holding down the **BOOT button**, press and hold down the **EN button** for 2 secs
3. Let go of the **EN button** while still holding the **BOOT button** for 2 secs, then let go of the **BOOT button**
4. Your board should now show up again in the “**Select Board**” toolbar afterwards

*\***BOOT and EN buttons** are labeled on the ESP32*



# ESP32 Boot Mode Example



## **SECTION III**

# **Review: ESP32 Functions**

# Digital Pin Functions

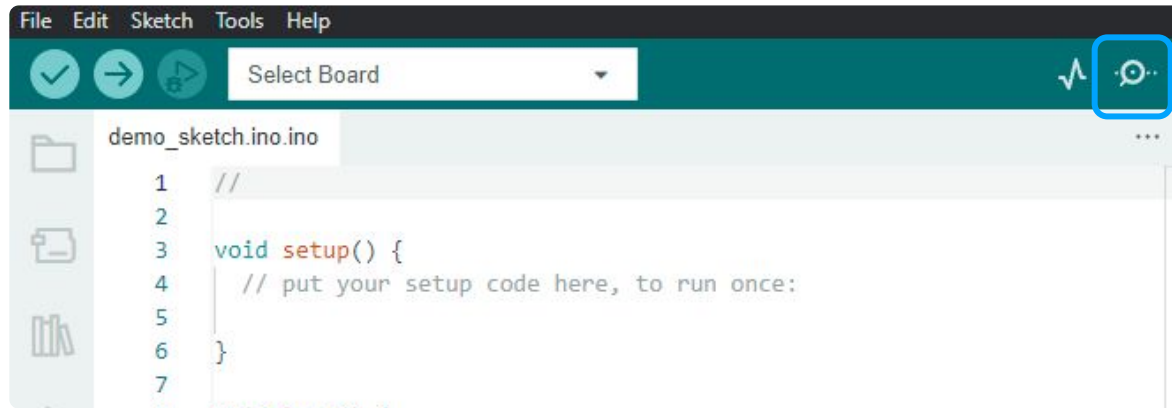


- `digitalWrite(int pin, int value)`
  - **Sets the voltage** at the output pin to either a **HIGH** (3.3V) or **LOW** (0V) value
  - Analogy - light switch and light bulb:
    - Like toggling a switch on and off
- `digitalRead(int pin)`
  - **Reads the voltage** at the input pin, returning **HIGH** (3.3V) or **LOW** (0V) as an integer (1 or 0)

# More Basic Functions

- `delay(int ms)`
  - **Pauses the program execution** by `ms` milliseconds
- `Serial.print("Message")`
  - Sends a string to the computer connected via USB and **displays the string on the Serial Monitor** in the IDE
- `Serial.println("Message")`
  - Sends a string to the computer connected via USB and **displays your string on the Serial Monitor** in the IDE, **followed by a newline**

# Using the Serial Monitor (Cont'd)



- In the absence of a debugger (the ESP32 is not capable of using one), **Serial.print** is an excellent tool to **help debug programs**
  - Print values to track across parts of your program
    - Unexpected values displayed to the Serial Monitor indicates an error

# Analog Pin Functions



- **analogWrite**(**int** pin, **int** value)
  - **Sets the average voltage** on digital output pin to a value in the **range 0–255** (0V to 3.3V)
  - Analogy - light dimmer:
    - You use the slide to set the bulb to anywhere *between* MAX brightness or MIN brightness
- **analogRead**(**int** pin)
  - **Reads the voltage** at the input pin, maps it to a value in the **range 0–4095** (0V to 3.3V) and returns that value
  - Use the aliases **A0**, **A1**, **A2...** for the pin number

## **SECTION IV**

# **ESP32 Serial Monitor + Potentiometer Exercise**

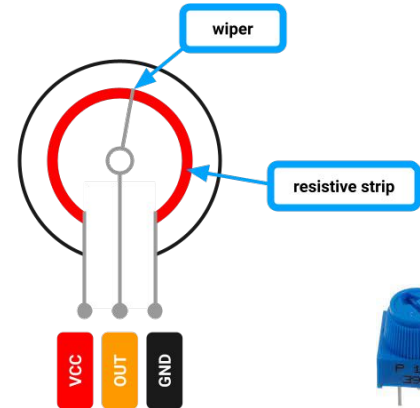
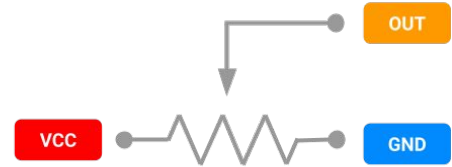


# What is a Potentiometer?

- A potentiometer is a *variable* resistor with 3 terminals: VCC, OUT, and GND
- We will use it as a voltage divider to only output a fraction of the supply voltage
  - This output pin voltage varies between the VCC and GND pin voltages based on the dial's position
- **Disclaimer: Don't turn the wiper too far past its limit (the knob is fragile and can break easily if turned too far)**



The positions of VCC and GND can be swapped

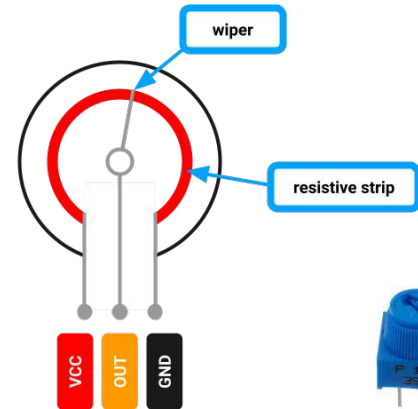
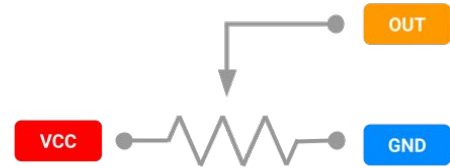


# What is a Potentiometer? (cont.)

- A potentiometer has many different applications, such as:
  - Volume control
  - Light dimming
  - Tuning and calibration
- **Trivia questions!**
  - If the wiper is all the way to the left, what is the voltage at the OUT pin?
  - If the wiper is all the way to the right, what would the value in Arduino IDE be between 0-4095?



The positions of VCC and GND can be swapped

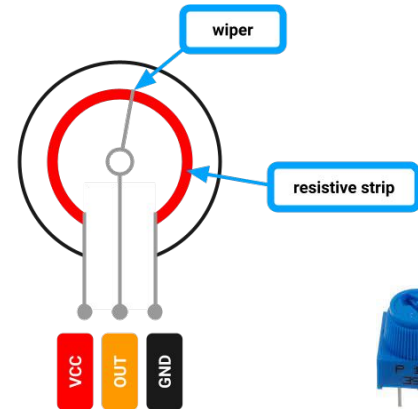
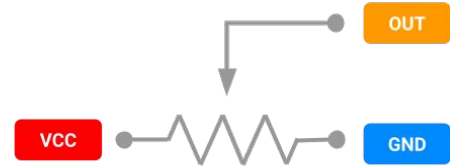


# What is a Potentiometer? (cont.)

- A potentiometer has many different applications, such as:
  - Volume control
  - Light dimming
  - Tuning and calibration
- **Trivia questions!**
  - If the wiper is all the way to the left, what is the voltage at the OUT pin?
  - Answer: **3.3V**
  - If the wiper is all the way to the left, what would the value in Arduino IDE be between 0-4095?
  - Answer: **4095**

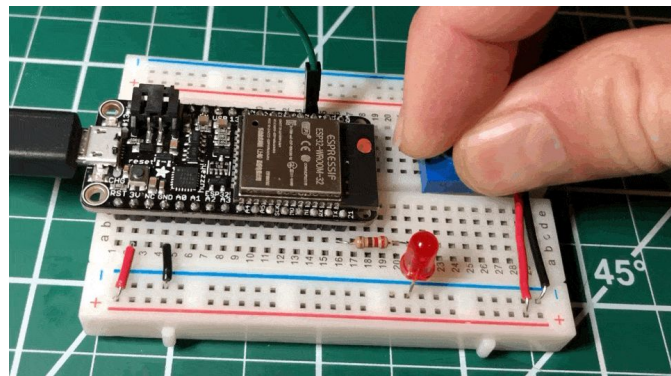


The positions of VCC and GND can be swapped



# Using Potentiometers with ESP32

- We can't just use `digitalRead()` to read values in between 0 - 5V off our potentiometer
- Instead, we'll be using `analogRead(int pin)`
  - The analog pin is wired to the ESP32's **analog-to-digital converter (ADC)**
    - Translates the analog signal to a discrete digital signal
- Now, let's look at some code showing this in action!



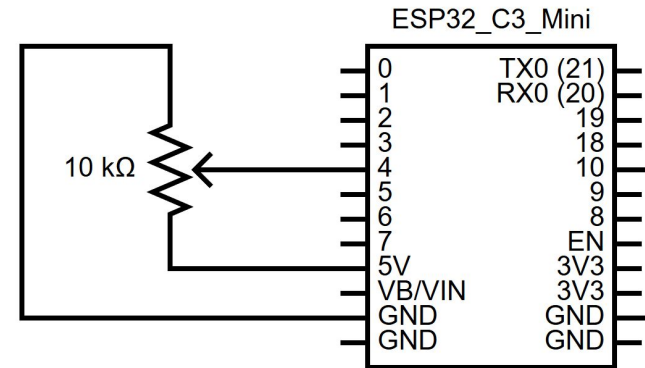
# LED Potentiometer Exercise

- Now, we're going to learn how to read values from a potentiometer!
- The goal of this exercise is to print the values read from a potentiometer output pin on the serial monitor
- To start: build the schematic you see on the right

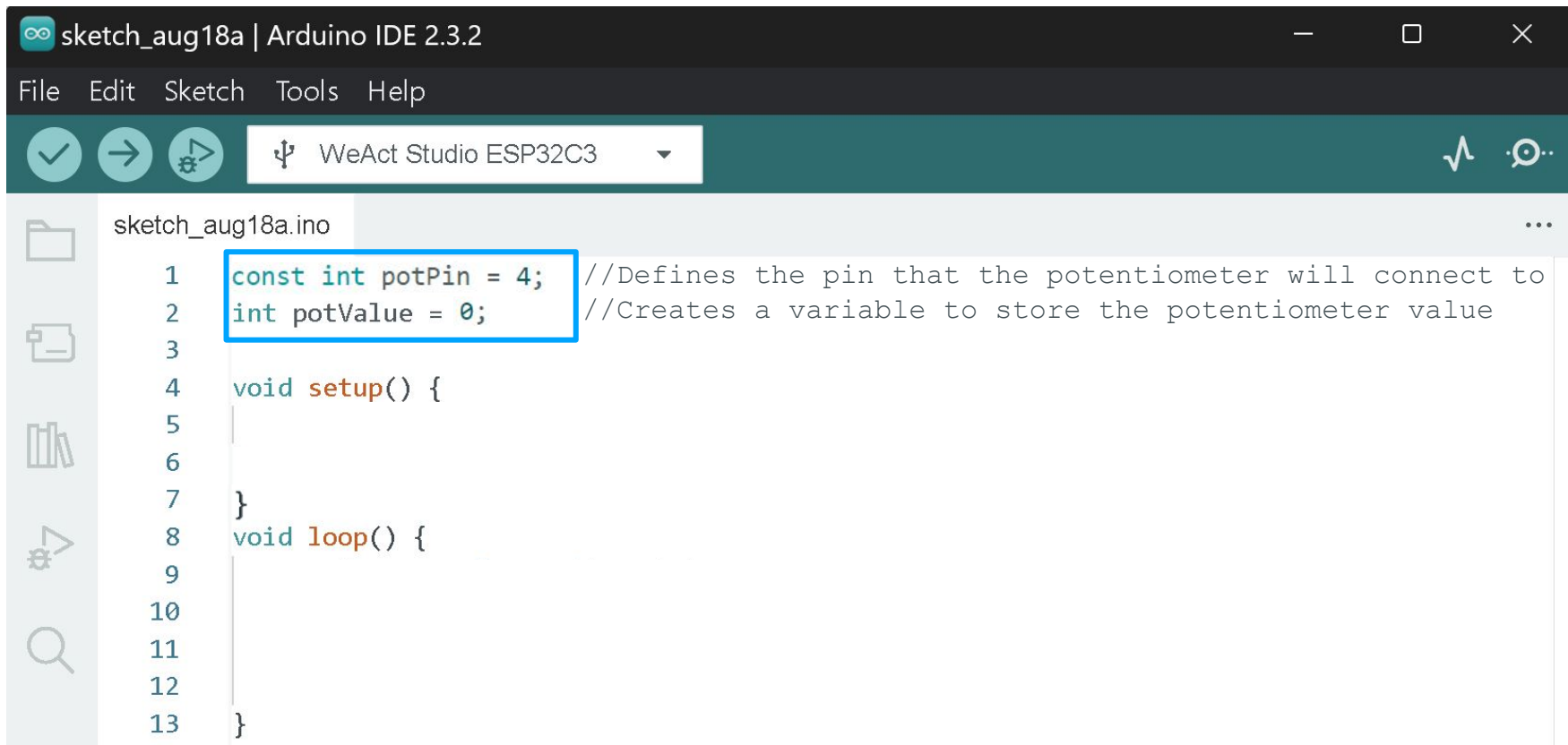


You will need:

- x1 ESP32
- x1 Breadboard
- x1 USB-A to USB-C Cable (For Programming)
- x1 Potentiometer (Blue)



# Potentiometer Code Example



```
sketch_aug18a.ino
1  const int potPin = 4; //Defines the pin that the potentiometer will connect to
2  int potValue = 0;    //Creates a variable to store the potentiometer value
3
4  void setup() {
5
6
7  }
8  void loop() {
9
10
11
12
13 }
```

# Potentiometer Code Example



```
sketch_aug18a | Arduino IDE 2.3.2
File Edit Sketch Tools Help
WeAct Studio ESP32C3

sketch_aug18a.ino
1  const int potPin = 4;
2  int potValue = 0;
3
4  void setup() {
5      Serial.begin(115200); // Begin serial communication at baud rate of 115200
6      pinMode(potPin, INPUT); // Sets the LED pin to output mode
7  }
8  void loop() {
9
10
11
12
13 }
```

# Potentiometer Code Example

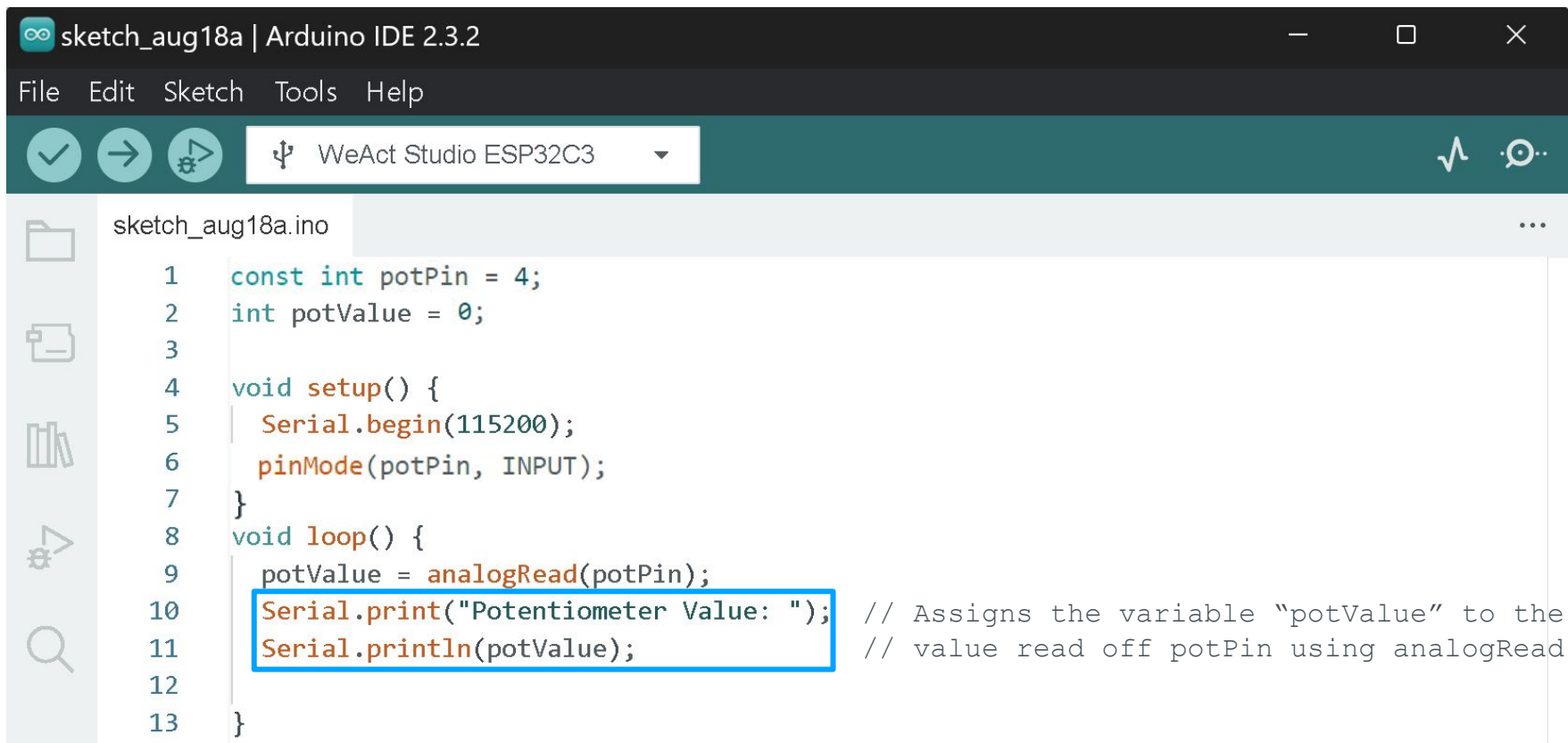


```
sketch_aug18a | Arduino IDE 2.3.2
File Edit Sketch Tools Help
WeAct Studio ESP32C3

sketch_aug18a.ino
1  const int potPin = 4;
2  int potValue = 0;
3
4  void setup() {
5      Serial.begin(115200);
6      pinMode(potPin, INPUT);
7  }
8  void loop() {
9      potValue = analogRead(potPin); // Assigns the variable "potValue" to the
10                                     // value read off potPin using analogRead
11
12
13 }
```



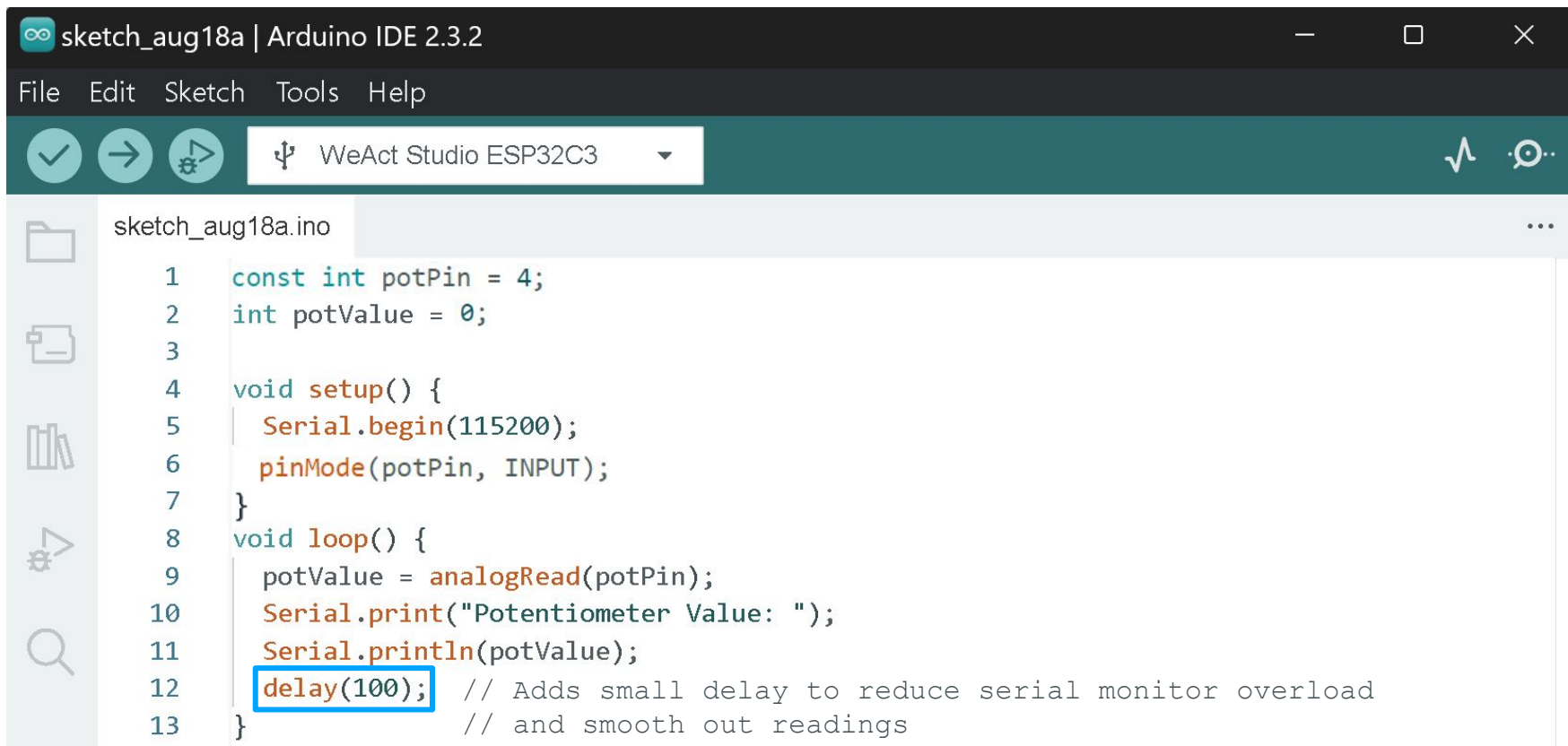
# Potentiometer Code Example



```
sketch_aug18a | Arduino IDE 2.3.2
File Edit Sketch Tools Help
WeAct Studio ESP32C3

sketch_aug18a.ino
1  const int potPin = 4;
2  int potValue = 0;
3
4  void setup() {
5      Serial.begin(115200);
6      pinMode(potPin, INPUT);
7  }
8  void loop() {
9      potValue = analogRead(potPin);
10     Serial.print("Potentiometer Value: "); // Assigns the variable "potValue" to the
11     Serial.println(potValue);             // value read off potPin using analogRead
12
13 }
```

# Potentiometer Code Example



sketch\_aug18a | Arduino IDE 2.3.2

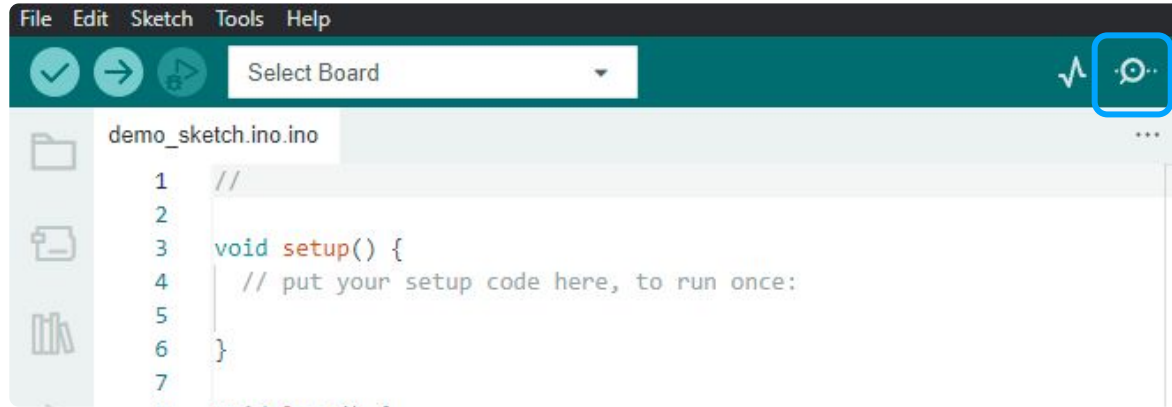
File Edit Sketch Tools Help

✓ → # WeAct Studio ESP32C3

sketch\_aug18a.ino

```
1  const int potPin = 4;
2  int potValue = 0;
3
4  void setup() {
5      Serial.begin(115200);
6      pinMode(potPin, INPUT);
7  }
8  void loop() {
9      potValue = analogRead(potPin);
10     Serial.print("Potentiometer Value: ");
11     Serial.println(potValue);
12     delay(100); // Adds small delay to reduce serial monitor overload
13 }              // and smooth out readings
```

# Using the Serial Monitor



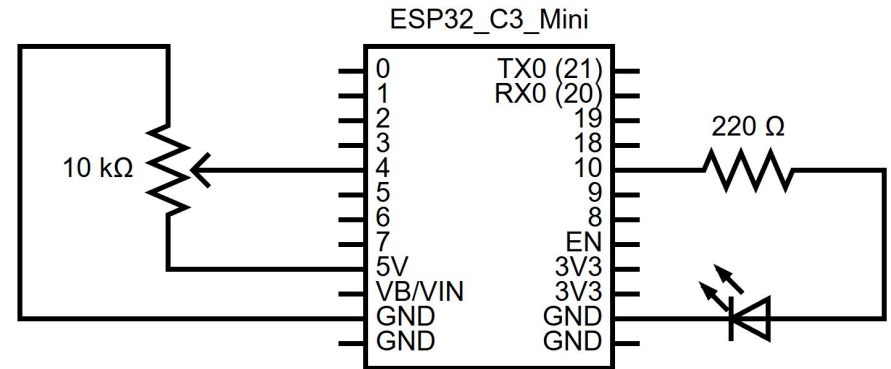
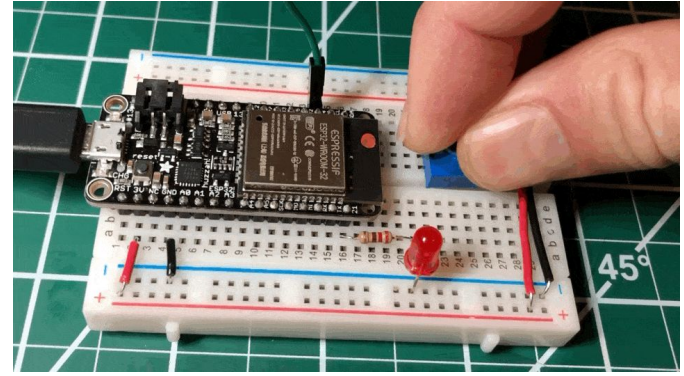
- While the ESP32 board is connected to the personal computer via USB, select **Serial Monitor** (the **magnifying glass** icon) in the IDE
  - A pane will appear at the bottom of the IDE window which displays all data sent by the ESP32 board using **Serial.print**

## **SECTION V**

# **LED Dimmer Exercise**

# LED Potentiometer Code Exercise

- Now, we're going to control the brightness of an LED with a potentiometer!
- The goal of this exercise is to make the LED grow brighter as you twist the potentiometer **clockwise**
- To start: build the schematic you see on the right

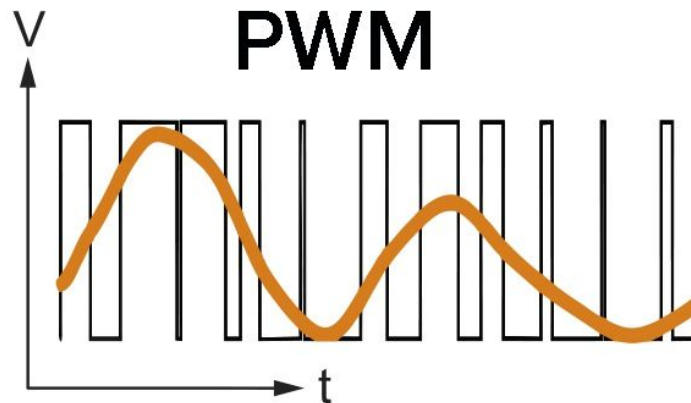


# Using Potentiometers with ESP32

- `digitalWrite()` can only set a pin's voltage to HIGH or LOW, nothing in between!
- We need to use a new function,

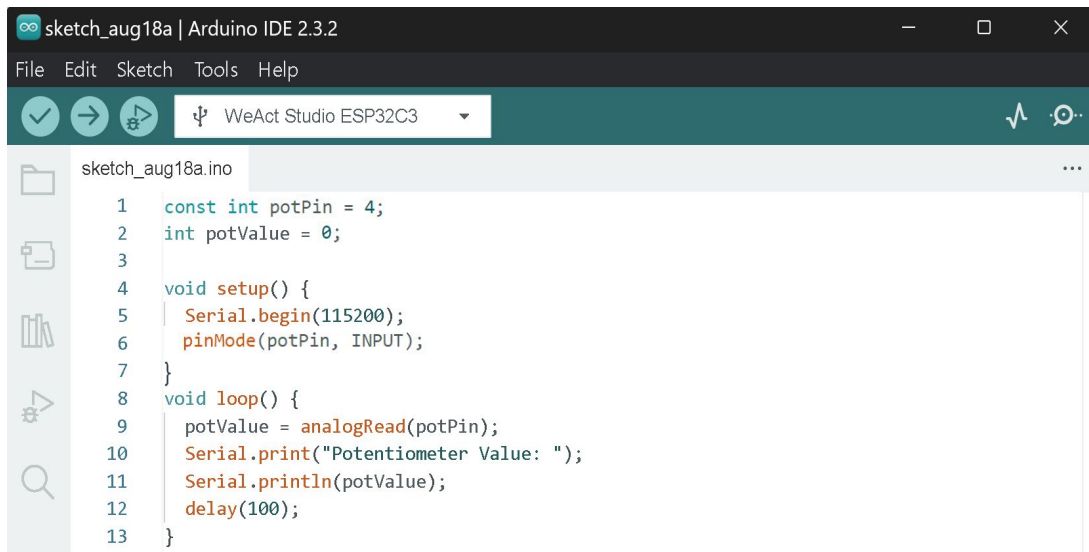
`analogWrite(int pin, int value)`

- With **pulse width modulation (PWM) waves**, we can generate voltages anywhere between 0V and 3.3V
- `int pin` - Reference a specific pin to use
- `int value` - Any value between 0 and 255 (Inputting 0 outputs 0V, and 127 outputs 1.65V, 255 outputs 3.3V, etc.)



# LED Potentiometer Code Exercise (cont.)

- Now that the schematic is finished, ensure that you have code to read the values of the potentiometer
- Feel free to reference the earlier example code if you need help starting!

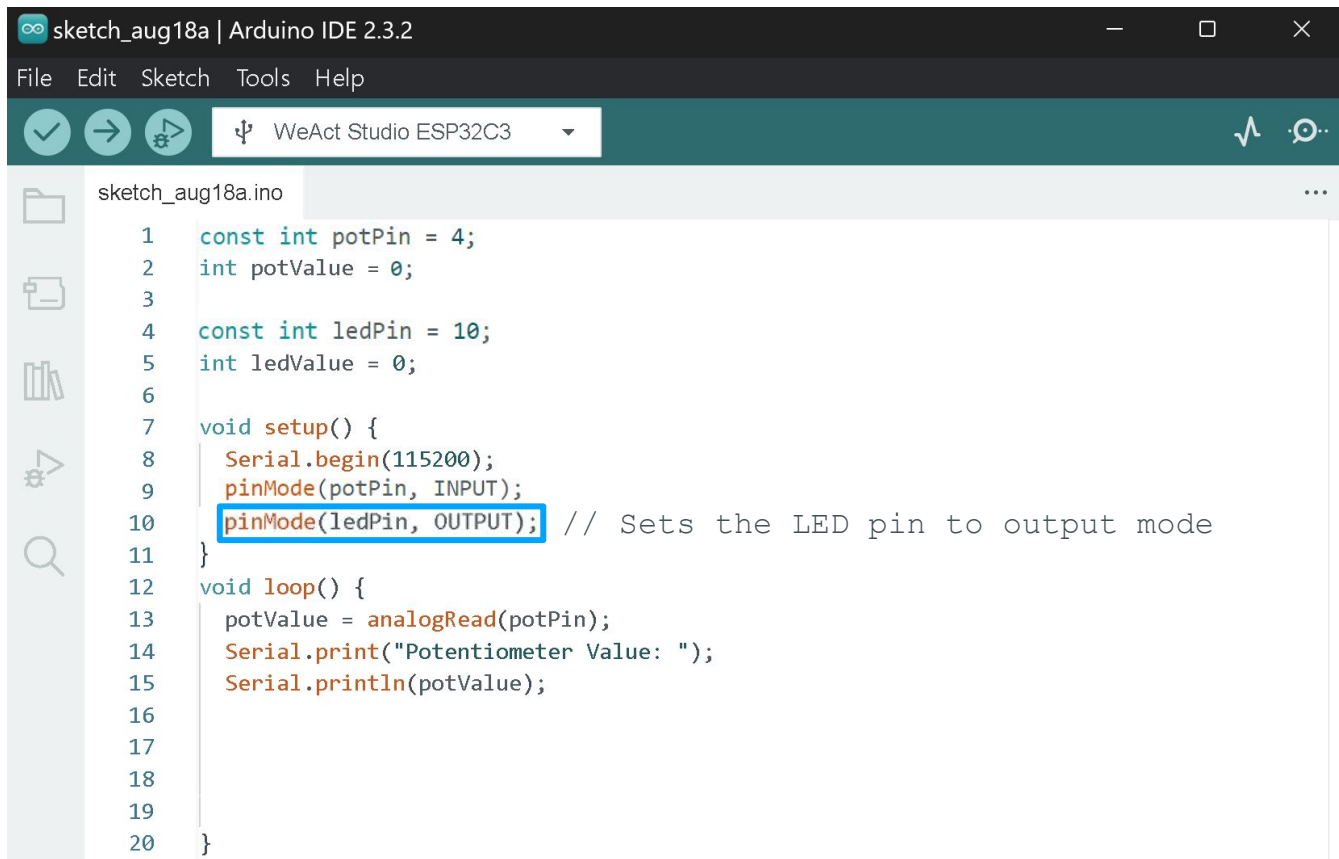


```
sketch_aug18a | Arduino IDE 2.3.2
File Edit Sketch Tools Help
[Icons] WeAct Studio ESP32C3
sketch_aug18a.ino
1  const int potPin = 4;
2  int potValue = 0;
3
4  void setup() {
5      Serial.begin(115200);
6      pinMode(potPin, INPUT);
7  }
8  void loop() {
9      potValue = analogRead(potPin);
10     Serial.print("Potentiometer Value: ");
11     Serial.println(potValue);
12     delay(100);
13 }
```

# LED Potentiometer Code Exercise (cont.)



# LED Potentiometer Code Exercise (cont.)



```
sketch_aug18a | Arduino IDE 2.3.2
File Edit Sketch Tools Help

[Icons] WeAct Studio ESP32C3 [Icons]

sketch_aug18a.ino ...

1  const int potPin = 4;
2  int potValue = 0;
3
4  const int ledPin = 10;
5  int ledValue = 0;
6
7  void setup() {
8      Serial.begin(115200);
9      pinMode(potPin, INPUT);
10     pinMode(ledPin, OUTPUT); // Sets the LED pin to output mode
11 }
12 void loop() {
13     potValue = analogRead(potPin);
14     Serial.print("Potentiometer Value: ");
15     Serial.println(potValue);
16
17
18
19
20 }
```

# LED Potentiometer Code Exercise (cont.)

sketch\_aug18a | Arduino IDE 2.3.2

File Edit Sketch Tools Help

✓ → ⚙

🔌 WeAct Studio ESP32C3

🔊 🔍

sketch\_aug18a.ino

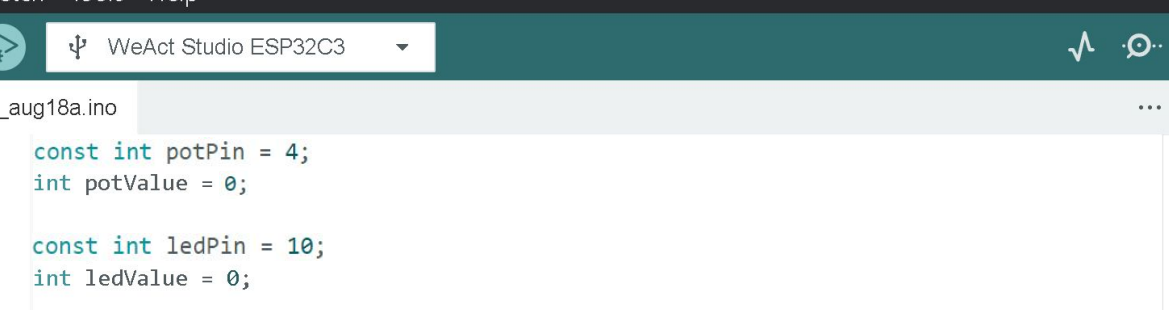
```
1  const int potPin = 4;
2  int potValue = 0;
3
4  const int ledPin = 10;
5  int ledValue = 0;
6
7  void setup() {
8      Serial.begin(115200);
9      pinMode(potPin, INPUT);
10     pinMode(ledPin, OUTPUT);
11 }
12 void loop() {
13     potValue = analogRead(potPin);
14     Serial.print("Potentiometer Value: ");
15     Serial.println(potValue);
16
17     ledValue = (potValue * 255) / 4095;
18
19
20 }
```

Assigns the variable “ledValue” to the potValue with some math:

We multiply by 255 and divide by 4095 because analogRead outputs a number from 0 to 4095, but the PWM range for analogWrite is only from 0 to 255, so we need to map values from the range of 0-4095 to 0-255 before writing them!

Ex. If the value is 1000 from the potentiometer, then  $1000 * 255 / 4095 \approx 62$

## LED Potentiometer Code Example (cont.)



The screenshot shows the Arduino IDE 2.3.2 interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for checking, running, and uploading code, along with a dropdown menu currently set to 'WeAct Studio ESP32C3'. The main workspace displays a sketch named 'sketch\_aug18a.ino' with the following code:

```
1  const int potPin = 4;
2  int potValue = 0;
3
4  const int ledPin = 10;
5  int ledValue = 0;
6
7  void setup() {
8      Serial.begin(115200);
9      pinMode(potPin, INPUT);
10     pinMode(ledPin, OUTPUT);
11 }
12 void loop() {
13     potValue = analogRead(potPin);
14     Serial.print("Potentiometer Value: ");
15     Serial.println(potValue);
16
17     ledValue = (potValue * 255) / 4095;
18     analogWrite(ledPin, ledValue); // Writes the stored value in ledValue to the
19     delay(100);                    // pin our LED is connected to
20 }
```

The line `analogWrite(ledPin, ledValue);` on line 18 is highlighted with a blue box.

# FAIR USE DISCLAIMER

Copyright Disclaimer under section 107 of the Copyright Act 1976, allowance is made for “fair use” for purposes such as criticism, comment, news reporting, teaching, scholarship, education and research.

Fair use is a use permitted by copyright statute that might otherwise be infringing.

Non-profit, educational or personal use tips the balance in favor of fair use.

## CC BY-NC-SA 4.0

This work by the Institute of Electrical and Electronics Engineers, UC Irvine Branch, is licensed under CC BY-NC-SA 4.0