

Topic	Motors	
Class Description	Students will learn about Motors and their types and develop an application on Servo Motors. Students will also learn how to use a Slide Potentiometer sensor as well.	
Class	PRO C261	
Class time	50 mins	
Goal	<ul style="list-style-type: none"> <li>• Motor Working and their types</li> <li>• Motor Connections</li> <li>• Writing program to develop application on motors using Slide Potentiometer sensor.</li> </ul>	
Resources Required	<ul style="list-style-type: none"> <li>• Teacher Resources:               <ul style="list-style-type: none"> <li>○ Laptop with internet connectivity</li> <li>○ Earphones with mic</li> <li>○ Notebook and pen</li> <li>○ Smartphone</li> </ul> </li> <li>• Student Resources:               <ul style="list-style-type: none"> <li>○ Laptop with internet connectivity</li> <li>○ Earphones with mic</li> <li>○ Notebook and pen</li> </ul> </li> </ul>	
Class structure	<b>Warm-Up</b> <b>Teacher-Led Activity</b> <b>Student-Led Activity</b> <b>Wrap-Up</b>	<b>10 mins</b> <b>15 mins</b> <b>15 mins</b> <b>10 mins</b>
Credit & Permissions:	Code samples used for Firebase-Google Authentication are licensed under the <a href="#">Apache 2.0 License</a> . Expo documentation used from - <a href="https://expo.io">https://expo.io</a> <b>Note: Keep this row section only if applicable</b>	

WARM-UP SESSION - 10 mins	
Teacher Action	Student Action
<p>Hey &lt;student's name&gt;. How are you? It's great to see you! Are you excited to learn something new today?</p> <p><b>Following are the WARM-UP session deliverables:</b></p> <ul style="list-style-type: none"> <li>Greet the student.</li> <li>Revision of previous class activities.</li> <li>Quizzes.</li> </ul>	<p><b>ESR:</b> Hi, thanks! Yes, I am excited about it!</p> <p>Click on the slide show tab and present the slides</p>
WARM-UP QUIZ Click on In-Class Quiz	
<p><b>Activity Details</b></p> <p><b>Following are the session deliverables:</b></p> <ul style="list-style-type: none"> <li>Appreciate the student.</li> <li>Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.</li> </ul>	
TEACHER-LED ACTIVITY 15mins	
Teacher Initiates Screen Share	
<ul style="list-style-type: none"> <li><b>Keypad connections.</b></li> <li><b>Algorithm to generate key codes randomly.</b></li> </ul>	
Teacher Action	Student Action
<p>So, in the last class, we learned how to use a keypad with ESP32 and created a guess key application on the same.</p> <p>Do you have any questions from the previous class?</p> <p><i>If the student has any doubts, clarify the doubts</i></p>	<p><b>ESR:</b> Varied</p>

<p>So let's start a new session.</p> <p>Are you excited?</p> <p>I will ask a few questions from you.</p> <p>Ready?</p>	<p><b>ESR: Yes!</b></p> <p><b>ESR: Yes!</b></p> <p><b>ESR: Yes!</b></p>
<p>Do you like racing cars, not only racing cars but any other car?</p> <p>How do they control speeds in cars?</p> <p>Have you seen Robots? How does it move?</p> <p>How does your indoor Fan move?</p> <p>How is movement, possible whether it is Rotary or sometimes linear too?</p> <p>The solution to all the above questions is Motor.</p> <p>Do you know what a motor is and how the motor works?</p> <p>In simple words, a motor is a device that changes a form of energy (<b>electrical energy</b>) into <b>mechanical energy</b> (Work) to produce motion. Here motion can be rotatory motion or linear motion depending upon the application. Every motor has a shaft that is used to connect with another device which will provide movement.</p> <p>The motor can be of various types:</p> <p>Stepper Motor</p>	<p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p> <p><b>ESR: Varied!</b></p>

<p>Servo motors</p> <p>AC motors</p> <p>DC motors</p> <p>Today we will focus on a very important and special motor that is a servo motor.</p> <p>We can see a lot of applications based on servo cars like</p> <ul style="list-style-type: none"> <li>• Robotics</li> <li>• Automatic Doors</li> <li>• Cameras</li> <li>• Vehicle</li> <li>• Solar Tracking</li> <li>• Miling</li> <li>• Radio Controlled toys</li> </ul>	
<p>Usually Servo motors rotate between 0 to 180 degrees. It is used to control the angular position of the object. The Servo Motor has three pins:</p> <p><b>VCC pin:</b> Connect this pin to <b>VCC (5V or 3.3v)</b>  <b>GND pin:</b> connect this pin to <b>GND (0V)</b>  <b>SIGNAL pin:</b> Use to control PWM of the signal</p> <ol style="list-style-type: none"> <li>1. Servo motors are controlled via a pulse width modulated (PWM) signal or using a servo library. Most servos rotate between 0° and 180° - starting and ending at fixed points relative to the motor. They accept pulses within a fixed range, commonly between 500 and 2500us.</li> <li>2. Pulse Width Modulation (PWM) is a technique of</li> </ol>	

<p>producing varying analog signals from a digital source.</p> <ol style="list-style-type: none"> <li>3. Digital signals can only be either HIGH or LOW, where the HIGH voltage is some fixed value depending on the circuit.</li> <li>4. On the other hand, an <b>analog</b> signal can be any voltage between HIGH and LOW. Normally, digital circuits can't freely vary voltage signals, but they can use PWM to get close enough. It works by repeatedly pulsing a HIGH digital signal on and off so that the <b>average</b> voltage coming from the circuit over time would be equivalent to an analog signal between HIGH and LOW.</li> <li>5. Servo motors are specialized for high response and accurate positioning.</li> </ol> <p>Let's see how it works on the simulator.</p>	
<p>Go to the <a href="#">wokwi</a> simulator and create a new ESP32 project,</p> <p>Let's try to create the circuit diagram first.</p> <p><b>Step -1: Select the components</b></p> <ul style="list-style-type: none"> <li>• 1 x <b>ESP32</b></li> <li>• 1 x <b>Servo</b>:</li> <li>• 1 x <b>Slider Potentiometer</b></li> </ul> <p><b>Step -2: Let's do the connections:</b></p>	

The circuit of this project consists of an **ESP32** Controller **Servo**

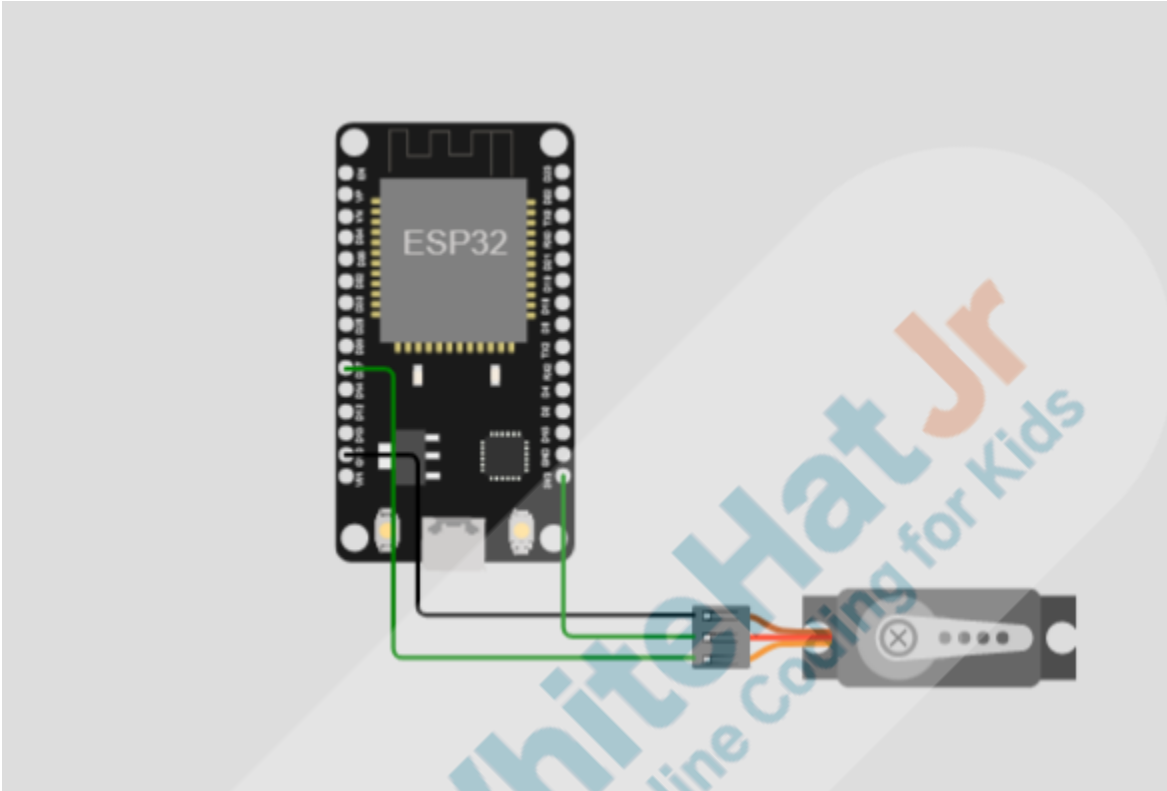
- Select **Servo** from the simulator list. Connect as per the instructions below :

SERVO	ESP32 PIN
VCC	VIN
GND	GND
PWM	D27

- Select the slider **potentiometer** from the simulator list. Connect as per the instructions below:

Potentiometer	ESP32 PIN
VCC	3v3
SIG	D27

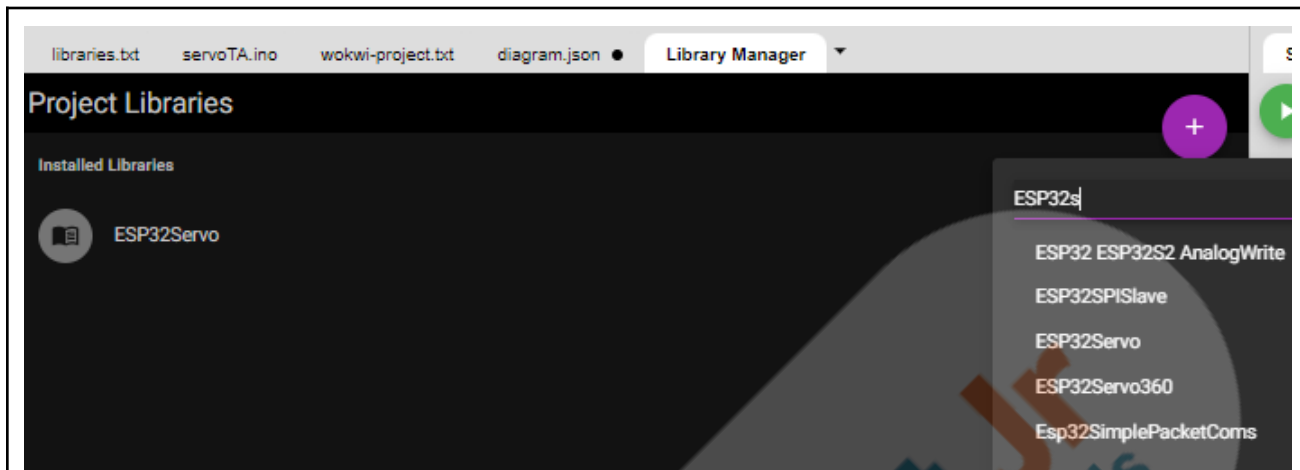
*Note: Wire color can be changed by **clicking over it** and selecting the color, or via the `diagram.json` file. Go to the `diagram.json` wire and change the color of the wire. Any design changes or color changes can be done via the `diagram.json` file. Keep the track of the components and change the design settings.*



Now it's time to write the code. For that, first, click on the Library manager and click on + sign and write **ESP32 Servo**

This way we can include the servo library.

**Note:** Perform the *same step on the student side too.*



Go to the **sketch.ino**, delete the entire code, and start writing our new code.

1. Include the **ESP32 Servo** library
2. Create **myservo** object
3. Define one variable, name them **pos** and assign a value.

```
#include <ESP32Servo.h>
Servo myservo;
int pos = 0;
```

Initialize using **void setup()** function

- **Serial.begin(115200)** is used to measure the speed of data exchange. This tells the Arduino to get ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second. That's 9600 binary ones or zeros per second and is commonly called a baud rate.
- **myservo.attach(pin)** is used to tell the controller that we are attaching a servo motor at the desired pin of the controller.

#### Parameters

- *myservo*: A variable of type **Servo**.
- *pin*: The number of the pin that the servo is attached



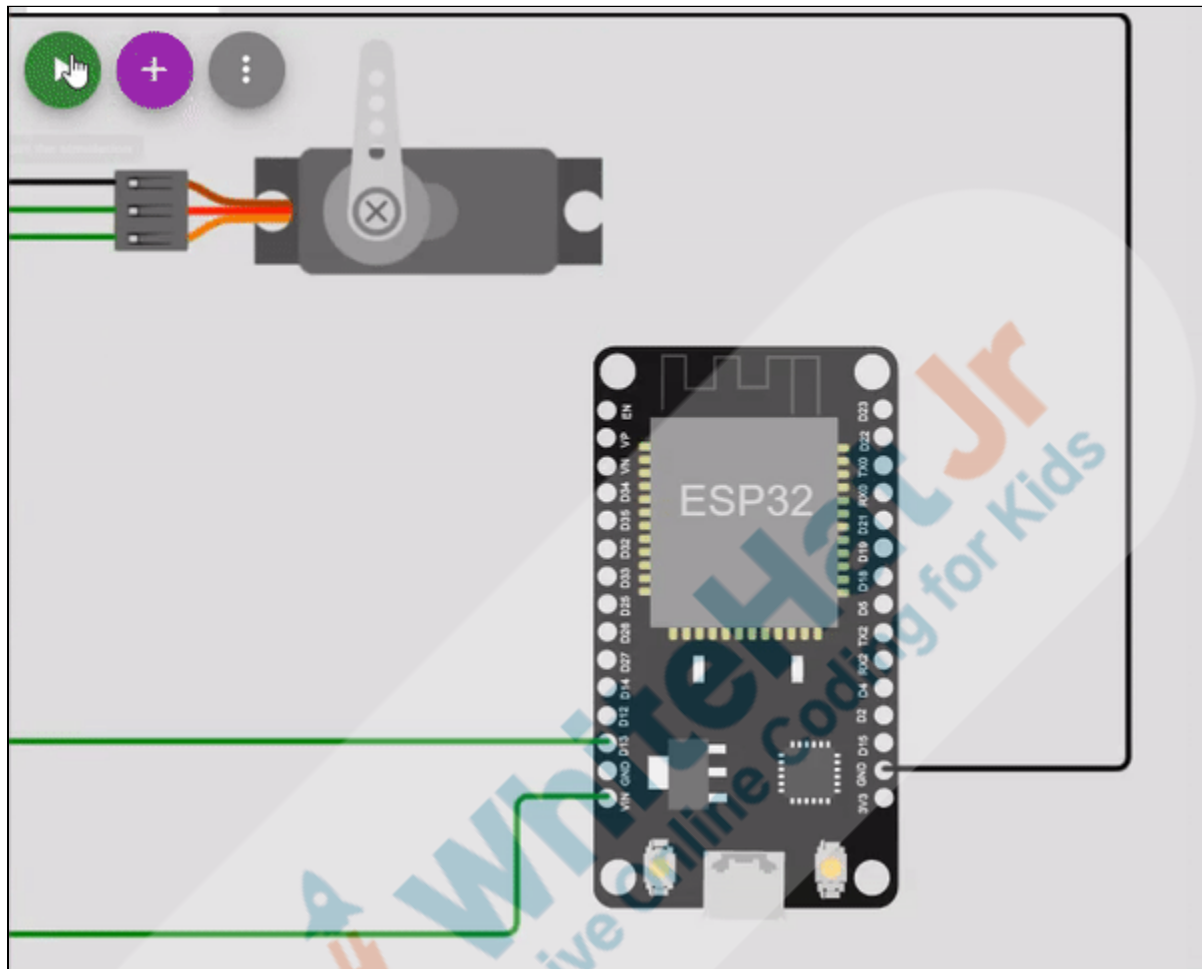
<pre>void setup() {   Serial.begin(115200);   Serial.println("Hello, ESP32!");   myservo.attach(27); }</pre>	
<p><b>Execution of the main process:</b></p> <ul style="list-style-type: none"> <li>• Now in the void loop we need to write the main process which needs to be processed.</li> <li>• <b>for loop</b> will help to increment &amp; decrement the motor rotation process. The servo motor rotates between 0 to 180. We need to set a for loop between 0 to 180.</li> <li>• For the forward position, we need to increment the same. But this time the starting position will be 0 and it will go till 180.</li> <li>• <b>myservo.write(pos)</b> tells servo to go to position in variable 'pos'</li> <li>• Set the delay to <b>15 ms</b></li> <li>• For the backward position, we need to decrease the same. But this time the starting position will be 180 and it will go till 0.</li> <li>• Set the delay to <b>15 ms</b></li> </ul>	

```
void loop() {  
  for (pos = 0; pos <= 180; pos += 1) {  
    myservo.write(pos);  
    delay(15);  
  }  
  for (pos = 180; pos >= 0; pos -= 1)  
    myservo.write(pos);  
    delay(15);  
}
```

**Output:**

- Click on the Save button and then click on the simulation button
- Press the key and see the output on the Serial Monitor of the simulator.
- Just press the keys and you will get the output.

The output for the following code will look like this.



### Student Stops Screen Share

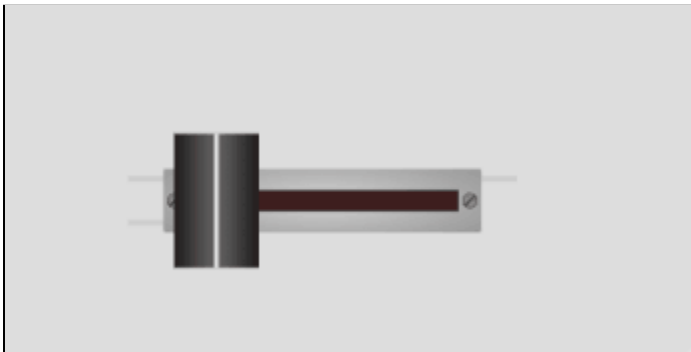
We have one more class challenge for you.  
Can you solve it?

Let's try. I will guide you through it.

### STUDENT-LED ACTIVITY- 15 mins

- Ask the student to press the ESC key to come back to the panel.
- Guide the student to start Screen Share.
- The teacher gets into Full Screen.

Student Initiates Screen Share	
<p style="text-align: center;"><u>ACTIVITY</u></p> <ul style="list-style-type: none"> <li>• <b>Servo and potentiometer connections.</b></li> <li>• <b>Writing algorithm for automatic barrier.</b></li> </ul>	
Teacher Action	Student Action
<p>I think it would be great if we can develop an application on the motor to understand better.</p> <p>Have you guys seen the barriers at society gates, which are controlled by the security guards?</p> <p>Those barriers are controlled with the help of a rope. How boring is that!</p> <p>Let's make an application, where the guard can control the opening and closing of that barrier with the help of a sensor. Also, he should be able to control the height or the opening of the barrier with the help of the sensor.</p> <p>For that, we are going to use a Slide potentiometer.</p> <p>Slide potentiometers are devices that are designed to change the output signal (towards the controller , ESP32) whenever someone presses and slides the <b>slider</b> over them.</p>	<p>Student open <a href="#">wokwi</a> simulator.</p> <p><b>ESR</b> : Yes</p>



You must have seen these kinds of devices in volume control, electronic devices etc.

Let's move to the coding part.

Go to the [wokwi](#) simulator and create a new ESP32 project,

Let's try to create the circuit diagram first.

#### Step -1: Select the components

- 1 x **ESP32**
- 1 x **Servo**:
- 1 x **Slider Potentiometer**

#### Step -2: Let's do the connections:

The circuit of this project consists of an **ESP32** Controller **Servo**

- Select **Servo** from the simulator list. Connect as per the instructions below:

<b>SERVO</b>	<b>ESP32 PIN</b>
--------------	------------------

<b>VCC</b>	<b>VIN</b>
<b>GND</b>	<b>GND</b>
<b>PWM</b>	<b>D27</b>

- Select the slider **potentiometer** from the simulator list. Connect as per the instructions below:

<b>Potentiometer</b>	<b>ESP32 PIN</b>
<b>VCC</b>	<b>3v3</b>
<b>SIG</b>	<b>D26</b>

*Note: Wire color can be changed by **clicking over it** and selecting the color, or via the `diagram.json` file. Go to the `diagram.json` wire and change the color of the wire. Any design changes or color changes can be done via the `diagram.json` file. Keep the track of the components and change the design settings.*

Let's write a program for the same.

Go to the **sketch.ino**, delete the entire code, and start writing our new code.

- Include the **ESP32 Servo** library
- Define variables for potentiometer and motors as **pot\_pin**, and **servo\_pin** along with their data type.

```
#include <ESP32Servo.h>

const byte servo_pin = 27;
const byte pot_pin = 26;
```

Initialize using **void setup()** function

- **Serial. begin(115200)** is used to measure the speed of data exchange. This tells the Arduino to get ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second. That's **115200** binary ones or zeros per second and is commonly called a baud rate.
- **servo.attach(servo\_pin)** is used to tell the controller that we are attaching a servo motor at the desired pin of the controller.
- **Parameters**
  - **servo**: a variable of type **Servo**
  - **pin**: the number of the pin that the servo is attached
- **servo.write()** is used to tell the servo motor to go to a particular angle.
- Set the **delay** of 1000 ms

```
void setup(){
    Serial.begin(115200);
    servo.attach(servo_pin);
    servo.write(0);
    delay(1000);
}
```

Execute the main process:

We need to read the input pin that can be possible using **analogRead()**

- define a variable **pot** using data type .

- **analogRead():** Reads the value from the specified analog pin. ESP32 contains a multichannel, 10-bit analog to digital converter. This means that it will map input voltages between 0 and the operating voltage(5V or 3.3V) into integer values between 0 and 1023. `analogRead()` read the input pin

Syntax: `analogRead(pin)`

- Parameter `pin`: the name of the analog input pin to read ,
- define variable **angle** using data type **int**. `angle` is the pot value re-scaled to 0-180.
- The resolution of the analog to digital converter can be done in this range (0-1023 for 10 bits or 0-4095 for 12 bits) using the `map()` function.
- **map method()** will scale servo angle with the potentiometer value .
- Set the **delay** of 10 ms

```
void loop(){
    int pot = analogRead(pot_pin);
    int angle = map(pot , 0 , 4095 , 0 , 180);
    servo.write(angle);

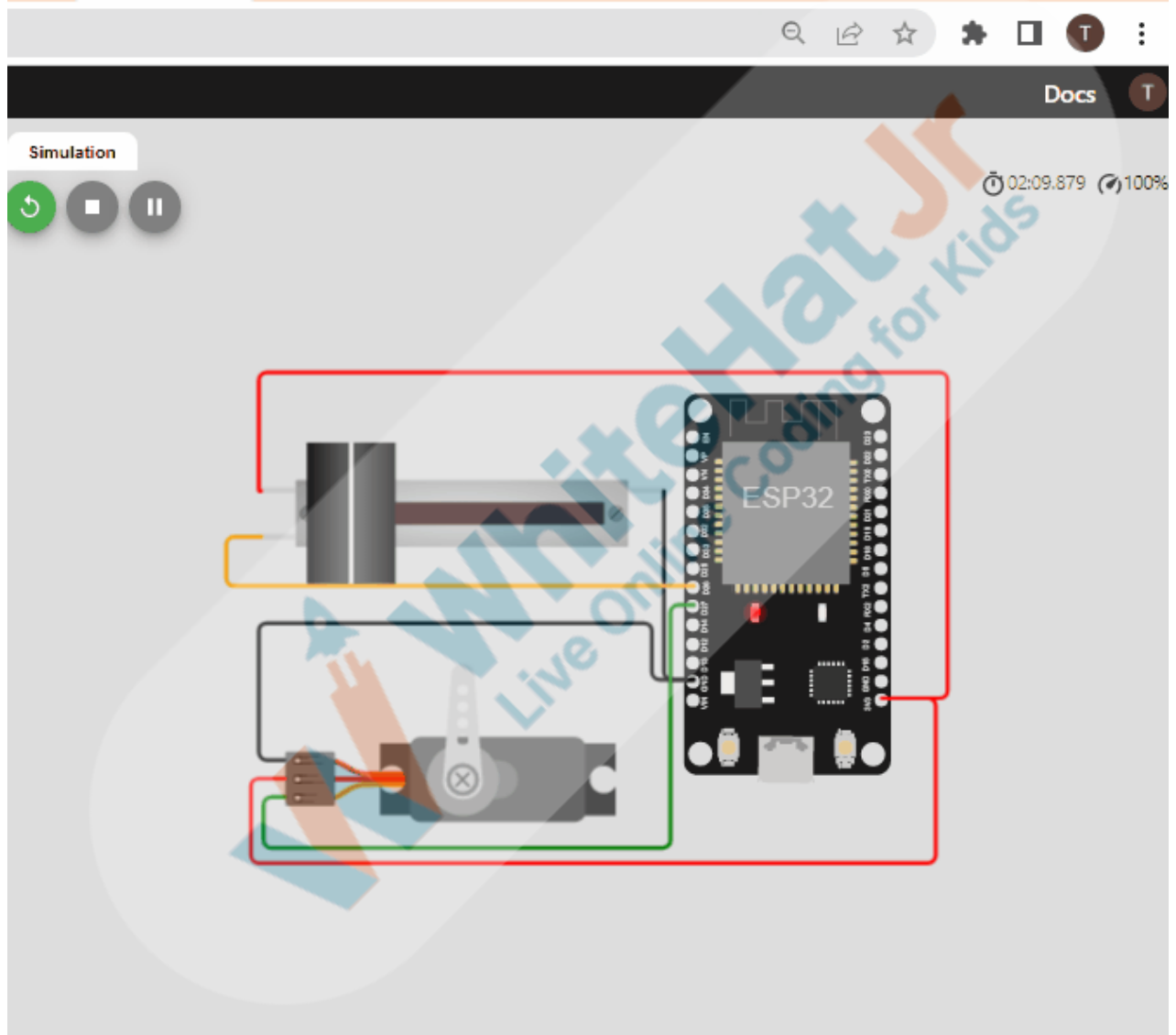
    // for better working of simulator
    delay(10);
}
```

#### Output:

- Click on the Save button and then click on the simulation button




- Press the key and see the output on the Serial Monitor of the simulator.
- Just press the keys and you will get the output.



Great, we learned how servo works.

Teacher Guides Student to Stop Screen Share	
WRAP-UP SESSION - 05 mins	
<b>Activity details</b>  <b>Following are the WRAP-UP session deliverables:</b> <ul style="list-style-type: none"> <li>• Appreciate the student.</li> <li>• Revise the current class activities.</li> <li>• Discuss the quizzes.</li> </ul>	
WRAP-UP QUIZ Click on In-Class Quiz	
<b>Activity Details</b>  <b>Following are the session deliverables:</b> <ul style="list-style-type: none"> <li>• Explain the facts and trivia</li> <li>• Next class challenge</li> <li>• Project for the day</li> <li>• Additional Activity (Optional)</li> </ul>	
FEEDBACK	
<ul style="list-style-type: none"> <li>• <b>Appreciate and compliment the student for trying to learn a difficult concept.</b></li> <li>• <b>Get to know how they are feeling after the session.</b></li> <li>• <b>Review and check their understanding.</b></li> </ul>	
Teacher Action	Student Action
<p>You get “hats-off” for your excellent work!</p> <p>In the next class, we will learn about servo motors.</p>	<p><i>Make sure you have given at least 2 hats-off during the class for:</i></p> <div> <div>Creatively Solved Activities +10</div> <div>Great Question +10</div> </div>

	<div>Strong Concentration  +10</div>
<b>PROJECT OVERVIEW DISCUSSION</b> Refer the document below in Activity Links Sections	
Teacher Clicks	<div>✕ End Class</div>
<b>ADDITIONAL ACTIVITIES</b> (Optional)	
Additional Activities	

ACTIVITY LINKS		
Activity Name	Description	Links
Teacher Activity 1	Simulator	<a href="https://wokwi.com/">https://wokwi.com/</a>
Teacher Activity 2	servo	<a href="https://docs.wokwi.com/parts/wokwi-servo">https://docs.wokwi.com/parts/wokwi-servo</a>
Teacher Reference 1	Teacher Activity Reference Code	<a href="https://github.com/procodingclass/261_SMTA.git">https://github.com/procodingclass/261_SMTA.git</a>
Teacher Reference 2	Student ActivityReference Code	<a href="https://github.com/procodingclass/261_PMSA.git">https://github.com/procodingclass/261_PMSA.git</a>
Teacher Reference 3	Project	<a href="https://s3-whjr-curriculum-uploads">https://s3-whjr-curriculum-uploads.</a>

		<a href="https://whjr.online/8fbb01f5-3e31-46eb-b5aa-b5ee278285c1.pdf">whjr.online/8fbb01f5-3e31-46eb-b5aa-b5ee278285c1.pdf</a>
Teacher Reference 4	Solution	<a href="https://github.com/procodingclass/PRO-C261-Project-Solution">https://github.com/procodingclass/PRO-C261-Project-Solution</a>
Teacher Reference 4	In_Class Quiz	<a href="https://s3-whjr-curriculum-uploads.whjr.online/d235b482-bc54-4ebe-852c-98151dba5dd3.pdf">https://s3-whjr-curriculum-uploads.whjr.online/d235b482-bc54-4ebe-852c-98151dba5dd3.pdf</a>
Student Activity 1	Simulator	<a href="https://wokwi.com/">https://wokwi.com/</a>
Student Activity 2	Keypad	<a href="https://docs.wokwi.com/parts/wokwi-servo">https://docs.wokwi.com/parts/wokwi-servo</a>
Student Activity 3	Student Activity Reference Code	<a href="https://github.com/procodingclass/PRO-C260-Student-Activity.git">https://github.com/procodingclass/PRO-C260-Student-Activity.git</a>