Topic	INTRODUCTION TO OLED(ORGANIC EMITTING LIGHT DIODE)	
Class Description	**Students will be introduced to the concept of OLED (Organic Light-Emitting device) and how its works and how OLED can be used to display text, shapes.**	
Class	**PRO C255**	
Class time	**50 mins**	
Goal	• OLED • Draw shapes on OLED • Draw text on OLED	
Resources Required	• Teacher Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Smartphone • Student Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up** **Teacher-Led Activity -1** **Student-Led Activity -2** **Wrap-Up**	**10 mins** **15 mins** **15 mins** **10 mins**
Credit & Permissions:	**Code samples used for Firebase-Google Authentication are licensed under the Apache 2.0 License.** **Expo documentation used from - https://expo.io** **Note: Keep this row section only if applicable**	

WARM-UP SESSION - 10 mins	
Teacher Action	**Student Action**
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** • Greet the student. • Revision of previous class activities. • Quizzes.	**ESR**: Hi, thanks! Yes, I am excited about it! Click on the slide show tab and present the slides
WARM-UP QUIZ Click on In-Class Quiz	
Activity Details **Following are the session deliverables:** • Appreciate the student. • Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.	
TEACHER-LED ACTIVITY-1 - 15mins	
Teacher Initiates Screen Share	
• **Display Text on OLED**	
Teacher Action	**Student Action**
Any doubts from the last class *If the student has any doubt, clarify the doubts.* Do you like to watch movies, cartoons, or sports?	 **ESR**: Varied!

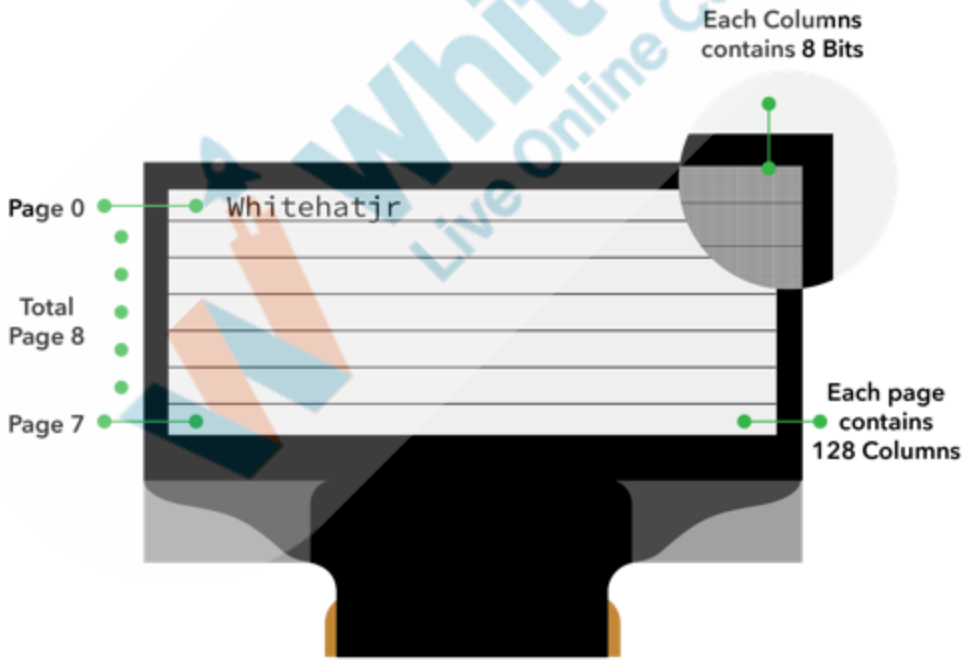
What device do you prefer to watch movies on? Is it your phone or laptop? The screen displays the picture, so have you noticed what kind of screen it is? How does it work? The screen can be of two types one is **LCD(Liquid Crystal Display)** and one is OLED **(Organic Light Emitting Device)** LCDs and OLEDs are the two most common display technologies for modern televisions. The vast majority of TVs are LCD, and only a few are OLED. The most common TV type is LCD, but OLED TVs offer better overall picture quality. Whenever mobile phone displays are discussed, OLEDs are used. This is similar, though not exactly the same, as the OLED tech found in TVs. Let's understand how OLED works? An OLED stands for an organic light-emitting diode. An OLED display is made up of pixels that glow when electricity is applied to them. It's like the heating elements in a toaster, but with less heat and a better resolution. This effect is called **electroluminescence** It is called organic because it is made up of organic substances, such as carbon.	**ESR**: Varied! **ESR**: Varied! **ESR**: Varied!
So OLED which we are using today is 128 pixels wide and 64 pixels tall, and each OLED is This tiny OLED helps to	

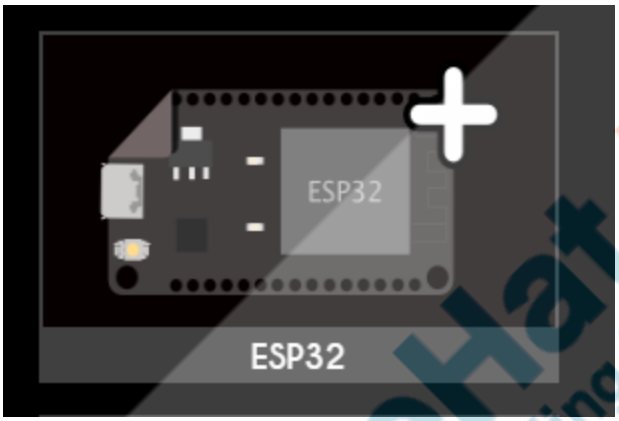
make you understand how OLED Works.

Knowing the memory map of your OLED display module will help you to understand how it displays.

This OLED has a 1K memory area that is organized into 8 pages (0 to 7). Each page has 128 columns/segments (blocks 0 to 127). Each column can store 8 bits of data (from 0 to 7).

Let's display **WHITEHATJR** on the OLED.



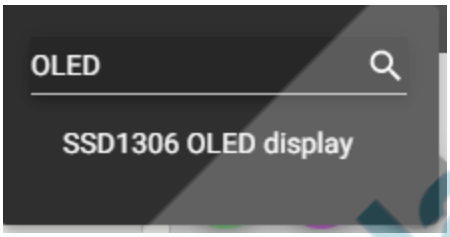
Teacher clicks on Teacher Activity 1	Studen click on Student Activity 1
Select **ESP32 board**	
	
To start, stop, and add new components, Click on the below buttons **Add component button** **Start Simulation** **Settings Property** Note: After clicking start simulation, two buttons will	

change into new below format  **Stop Simulation**  **Pause simulation**	
To save, rename, delete, create files, and upload files **Click on the small triangle icon** **next to Library** **Delete**: Select the file by clicking on the file and then click on the **small triangle icon** **to delete a particular file.** **Create:** Click on the **small triangle icon** to create a new file. Rename: Select the file by clicking on the file and then click on the **small triangle icon** **to rename a particular file.** **Save**: After renaming, Click on **Save** Button, it will automatically save your program in your Project list. **Download**: To save it on your computer you can directly **download Project zip** option Upload: Click on the **small triangle icon** and select the **Upload file(s)** option. *Note: We can upload multiple files once while uploading*	

code but the circuit diagram needs to be designed every time.

Step -1:Select the material from the Simulator

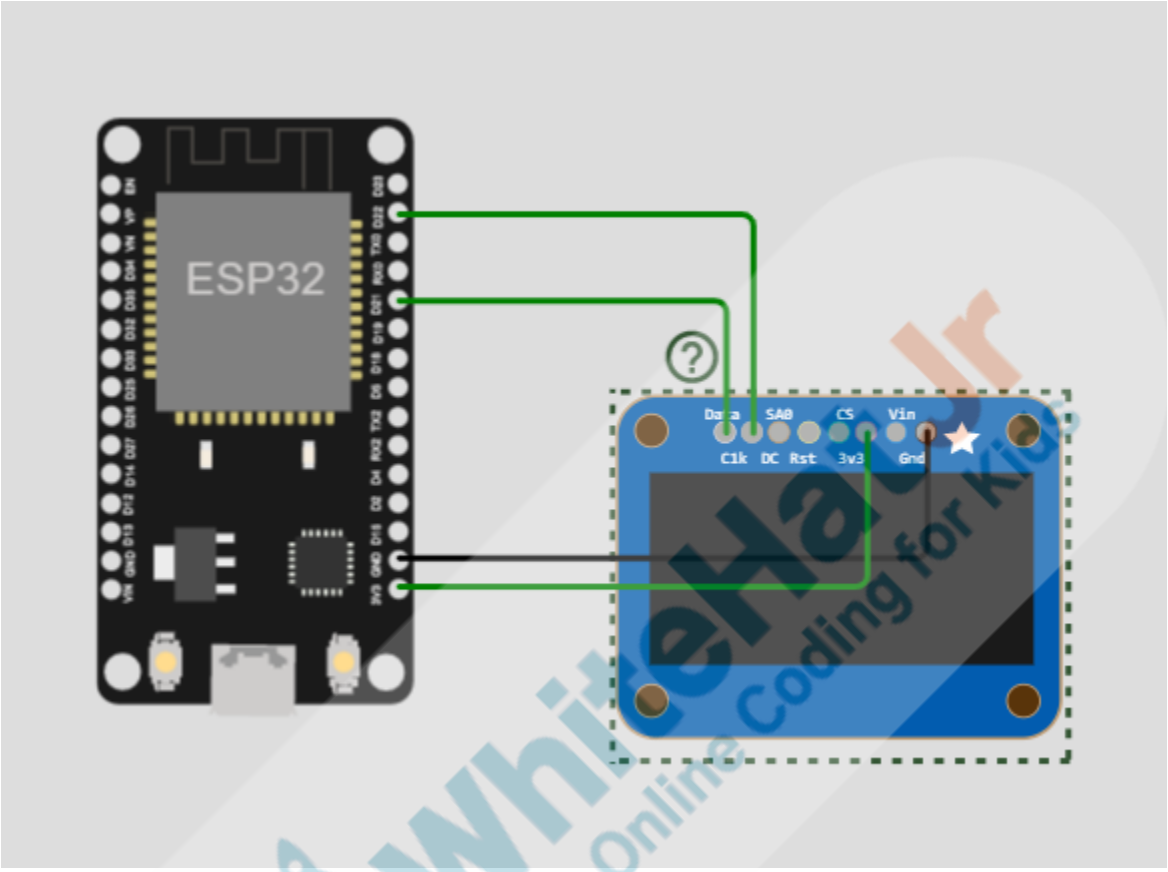
- 1 x ESP32
- 1 x OLED Click on + Sign and selects OLED



Step -2: Let's do connections:

- Insert OLED into the breadboard
- Take four jumper wires.

OLED PIN	ESP32 PIN
VCC	**3.3 V**
GND	**GND**
Data	GPIO **21**
CLK	GPIO **22**



To control the **OLED** display, we need to install libraries

1. Click on the small triangle icon next to Library Manager
2. Select New File
3. Name the file libraries.txt
4. Write down **Adafruit SSD1306**

Note: Follow the same step at student systems too. For reference, everything is mentioned again in the student activity.

sketch.ino | diagram.json ● | libraries.txt ● | Library Manager

```
# Wokwi Library List
# See https://docs.wokwi.com/guides/libraries

Adafruit SSD1306

```

Let's write the Program:

- **SPI.h Serial Peripheral Interface (SPI)** is a synchronous serial communication protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances. When using SPI, there is always one master device (usually a microcontroller) that controls all peripheral devices.
- **Wire.h** This **library** allows you to communicate with I2C / devices. I2C is **a serial communication protocol**, so data is transferred bit by bit along a single wire.
- **Adafruit_GFX.h:** This library offers a common graphical syntax and set of functions for all LCD displays, OLED displays, and LED matrices.
- **Adafruit_SSD1306** : This library takes care of low-level communication with the hardware.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

Define **SCREEN_WIDTH & SCREEN_HEIGHT** for OLED
Our **OLED** size is a 1**28×64**

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

Declaration of an **SSD1306** display that connects to **I2C** communication using **Wire** Library

- Initialize a **display** object with the **SCREEN_WIDTH & SCREEN_HEIGHT** defined earlier with the I2C communication protocol.
- A value of **(-1)** indicates that our OLED display does not have a **RESET** pin. Sometimes OLED displays have a RESET pin on the OLED, in that case, we should connect it to a GPIO and should include the GPIO number as a parameter.

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

Initialize using **void setup()** function

- Serial.begin(115200): Sets the **data rate** in bits per second (baud) for **serial** data transmission.
- Initialize the OLED display with the begin() method.
- If the OLED displays nothing, check the OLED address at **0x3C**. In our case, the address is 0x3C.
- If we are not able to connect to the display, it prints a message on the Serial Monitor.
- If something fails, don't proceed further, try to repeat the process using **for() loop**

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

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
```

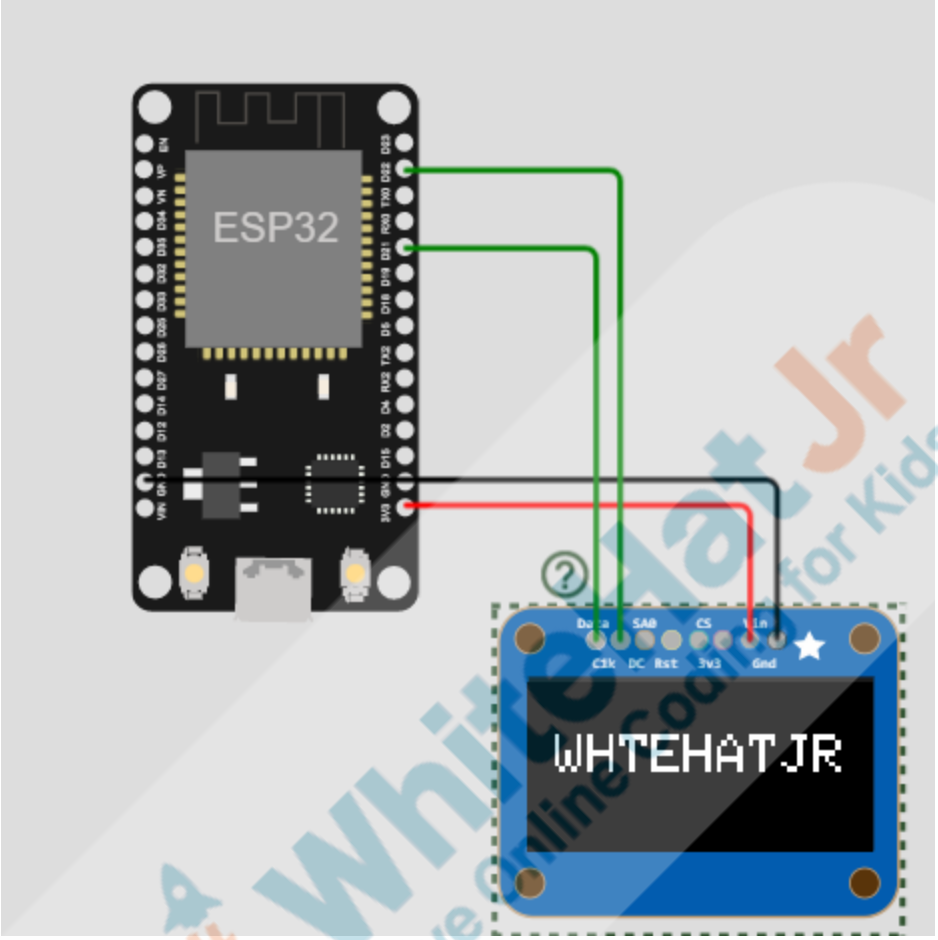
Print data on OLED

1. In order to allow the OLED to initialize, add a **two-second delay** before writing text
2. Clear the display buffer with the **clearDisplay()** method after initializing the display
3. To **write** text, you must first set the font size, color, and location where the text will be displayed in the OLED and the data which needs to be printed.
4. Set the font size using the **setTextSize()** method
5. Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
6. Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(0,10)**.
7. As a final step, you can send the text to the display using the **println()** method, as follows:
8. At last, we need to call the **display()** method to actually display the text on the screen.

```
    delay(2000);
    display.clearDisplay();

    display.setTextSize(2);
    display.setTextColor(WHITE);
    display.setCursor(10, 20);
    display.println("WHTEHATJR");
    display.display();
}
```

Call the main function using **void loop()**	
`` void loop() { } ```	
Output: Click on the Save button and then click on the simulation button *Note:* *If your OLED display is not showing anything:* *Check that the OLED display is properly wired* *Output will look like this:*	



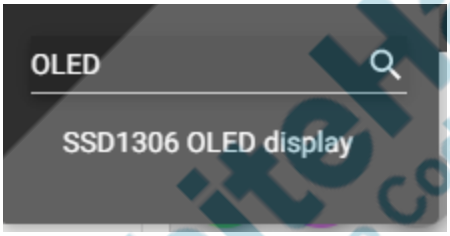
Student Stops Screen Share	
Please share your screen with me.	
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.	
STUDENT-LED ACTIVITY-2 - 15 mins	
• Ask the student to press the ESC key to come back to the panel. • Guide the student to start Screen Share.	

- The teacher gets into Full Screen.

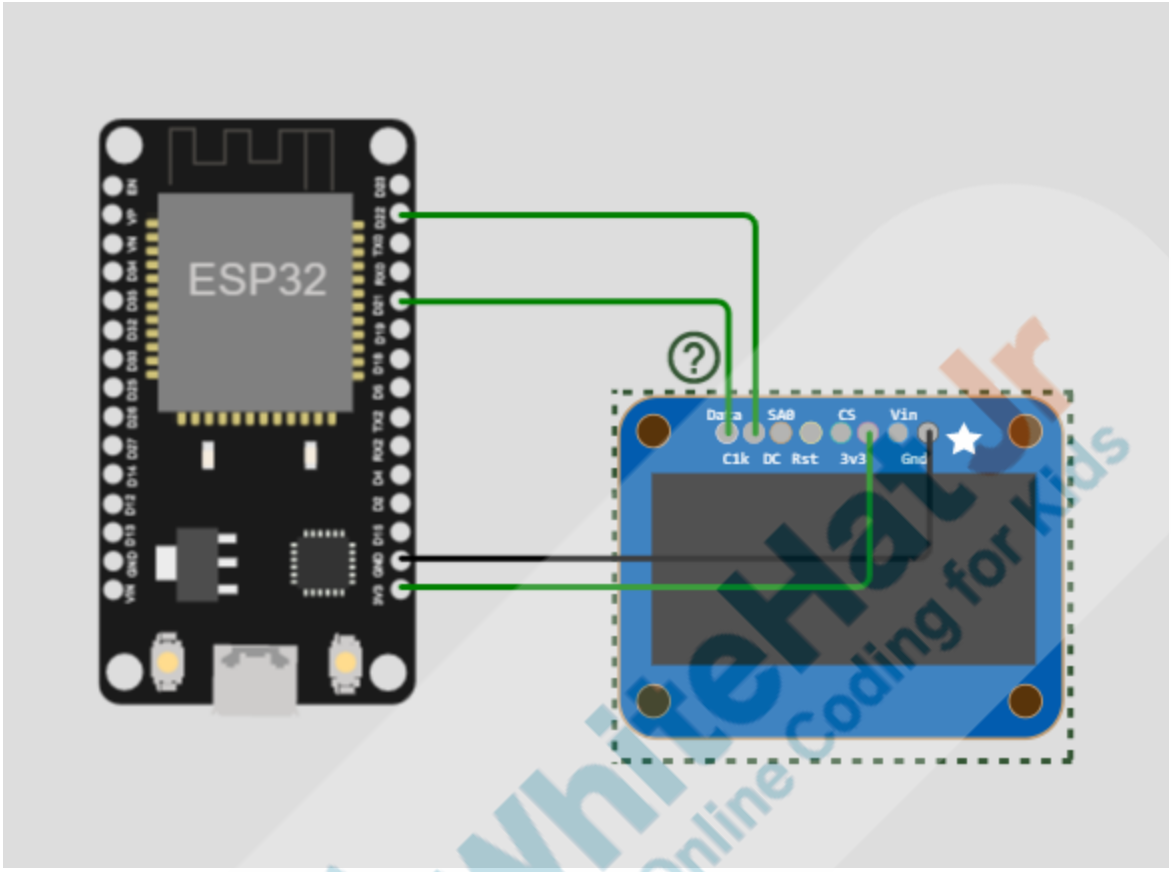
Student Initiates Screen Share

ACTIVITY

- Display Patterns on OLED

Teacher Action	Student Action
Step -1:Select the material from the Simulator • 1 x ESP32 • 1 x OLED Click on + Sign and select OLED	
	
Step -2: Let's do connections: • Insert OLED into the breadboard • Take four jumper wires.	

OLED PIN	ESP32 PIN
VCC	**3.3 V**
GND	**GND**
Data	GPIO **21**
CLK	GPIO **22**



To control the **OLED** display, we need to install libraries

5. Click on the small triangle icon next to Library Manager
6. Select New File
7. Name the file libraries.txt
8. Write down **Adafruit SSD1306**

sketch.ino | diagram.json ● | libraries.txt ● | Library Manager

```
# Wokwi Library List
# See https://docs.wokwi.com/guides/libraries

Adafruit SSD1306

```

Import Libraries:

- **SPI.h Serial Peripheral Interface (SPI)** is a synchronous serial communication protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances.When using SPI, there is always one master device (usually a microcontroller) that controls all peripheral devices.

- **Wire.h** This **library** allows you to communicate with I2C / devices. I2C is **a serial communication protocol**, so data is transferred bit by bit along a single wire.

- **Adafruit_GFX.h:** This library offers a common graphical syntax and set of functions for all LCD displays, OLED displays, and LED matrices.

- **Adafruit_SSD1306** : This library takes care of low-level communication with the hardware.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

Define **SCREEN_WIDTH & SCREEN_HEIGHT** for OLED
Our **OLED** size is a 1**28×64**

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

Declaration of an **SSD1306** display that connects to **I2C** communication using **Wire** Library

- Initialize a **display** object with the **SCREEN_WIDTH & SCREEN_HEIGHT** defined earlier with I2C communication protocol.
- A value of **(-1)** indicates that our OLED display does not have a **RESET** pin. Sometimes OLED displays have a RESET pin on the OLED, in that case we should connect it to a GPIO and should include the GPIO number as a parameter.

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

Initialize using **void setup()** function

- Serial.begin(115200): Sets the **data rate** in bits per second (baud) for **serial** data transmission.
- Initialize the OLED display with the begin() method .
- If the OLED displays nothing, check the OLED address at **0x3C**. In our case, the address is 0x3C.
- If we are not able to connect to the display, it prints a message on the Serial Monitor.
- If something fails, don't proceed further, try to repeat the process using **for() l**oop
- Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(0,0).**

```
void setup() {
  Serial.begin(9600);

  // initialize OLED display with I2C address 0x3C
  if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("failed to start SSD1306 OLED"));
    while (1);
  }

  delay(2000); // wait two seconds for initializing
  oled.setCursor(0, 0);
}
```

Now, it's time to write a code for shapes that need to display under the main function i.e **void loop()** function.

Circle:

1. Clear the display buffer with the **clearDisplay()** method after initializing the display
2. **drawCircle** method is used to draw circle shape on the OLED**.** drawCircle will use **X** and Y coordinates along with Radius (CenterX, CenterY, Radius in Pixels, WHITE);
3. **display.display()** is used to apply the changes.
4. Set a **delay** of 1s
5. **fillCircle** method is used to fill color in the circle shape on the OLED**. fillCircle** will use **X** and Y coordinates along with Radius (CenterX, CenterY, Radius in Pixels, WHITE);

```
void loop() {
  // draw a circle
  display.clearDisplay();
  display.drawCircle(50, 30, 30, WHITE);
  display.display();
  delay(1000);

  // fill a circle
  display.clearDisplay();
  display.fillCircle(50, 30, 30, WHITE);
  display.display();
  delay(1000);
```

Triangle:

1. Clear the display buffer with the **clearDisplay()** method after initializing the display.

2. **drawTriangle()** method is used to draw a triangle shape on the OLED**. drawTraiangle()** will use **X** and Y coordinates for three sides of a triangle along with color.

3. **display.drawTriangle (FirstX , FirstY, SecondX, SecondY, ThirdX, ThirdY, WHITE).**

4. **display.display()** is used to apply the changes.

5. Set a **delay** of 1s.

6. **fillTriangle()** method is used to fill color in a triangle shape on the OLED**. drawtriangle()** method will use **X** and Y coordinates for three sides of a triangle along with color.

```
// draw a triangle
display.clearDisplay();
display.drawTriangle(50, 10, 0, 60, 60, 60, WHITE);
display.display();
delay(1000);

// fill a triangle
display.clearDisplay();
display.fillTriangle(50, 10, 0, 60, 60, 60, WHITE);
display.display();
delay(1000);
```

Rectangle

Clear the display buffer with the **clearDisplay()** method after initializing the display

drawRectangle() method is used to draw a rectangle shape on the OLED**. drawRectangle** will use **X** and Y coordinates, Width & Height in Pixels along with the color

display.drawRect(StartX, StartY, Width in Pixels, Height in Pixels, WHITE);

display.display() is used to apply the changes.

Set a **delay** of 1s

fillRectangle() method is used to fill color in a rectangle shape on the OLED**. drawRectangle** will use **X** and Y coordinates, Width & Height in Pixels along with the color

```
  // draw a rectangle
  display.clearDisplay();
  display.drawRect(40, 20, 60, 40, WHITE);
  display.display();
  delay(1000);

  // fill a rectangle
  display.clearDisplay();
  display.fillRect(40, 20, 60, 40, WHITE);
  display.display();
  delay(1000);

}
```

Output:

Compile and upload the program to the ESP32 board using Arduino IDE

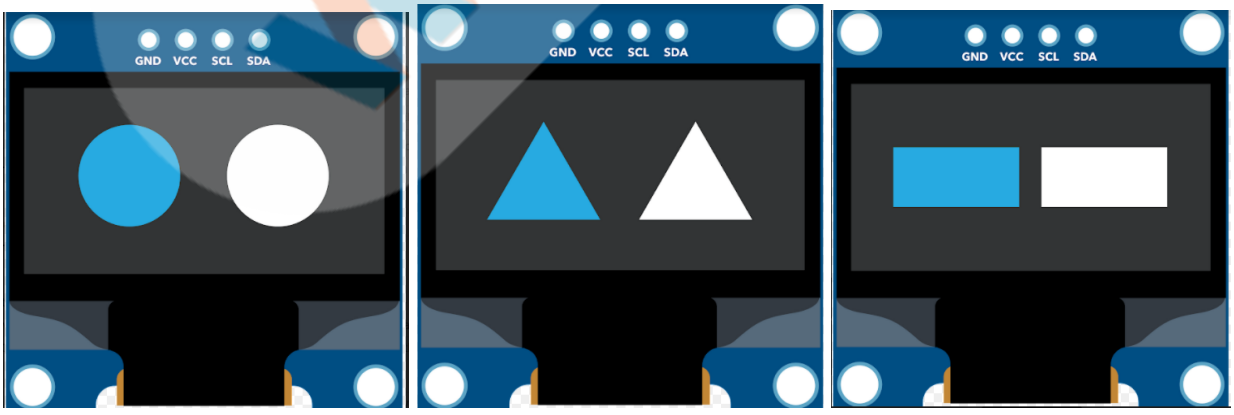
- Verify the program by clicking the Tick option.
- Upload the program by clicking the arrow option.

Note:

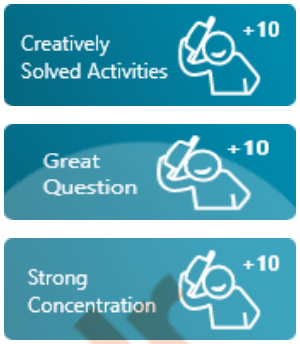
If the port is not selected, insert the USB cable into Computer's port and select the port

If your OLED display is not showing anything:

Check that the OLED display is properly wired



So, today we learned about OLED and how to display text and patterns on OLED. That's fun!	
Teacher Guides Student to Stop Screen Share	
WRAP-UP SESSION - 05 mins	
Activity details **Following are the WRAP-UP session deliverables:** ● Appreciate the student. ● Revise the current class activities. ● Discuss the quizzes.	
WRAP-UP QUIZ Click on In-Class Quiz	
Activity Details **Following are the session deliverables:** ● Explain the facts and trivia ● Next class challenge ● Project for the day ● Additional Activity (Optional)	
FEEDBACK ● **Appreciate and compliment the student for trying to learn a difficult concept.** ● **Get to know how they are feeling after the session.** ● **Review and check their understanding.**	
Teacher Action	**Student Action**
You get "hats-off" for your excellent work!	*Make sure you have given at least 2 hats-off during the class for:*

In the next class, we will learn about web servers	
PROJECT OVERVIEW DISCUSSION Refer the document below in Activity Links Sections	
Teacher Clicks 	
ADDITIONAL ACTIVITIES (Optional)	
Additional Activities	

ACTIVITY LINKS		
Activity Name	**Description**	**Links**
Teacher Activity 1	Reference Code -Text	https://github.com/procodingclass/PRO-C255-Reference-Code-TA

Teacher Activity 2	Reference Code -Shapes	https://github.com/procodingclass/PRO-C255-Student-Reference-Code-
Teacher Reference 1	Project	https://s3-whjr-curriculum-uploads.whjr.online/abeee780-50a9-4217-89d5-1e26ad4dcde3.pdf
Teacher Reference 2	Project Solution	https://github.com/procodingclass/PRO-C254-Project-Solution.git
Teacher Reference 3	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/66af2d7d-283c-4009-8f3d-414eec1e79fe.pdf