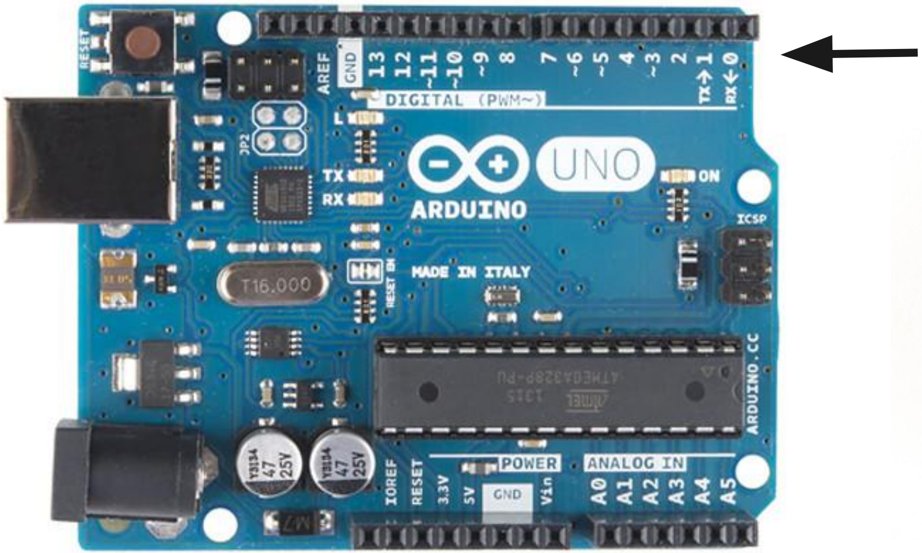
Tinkering Boot Camp

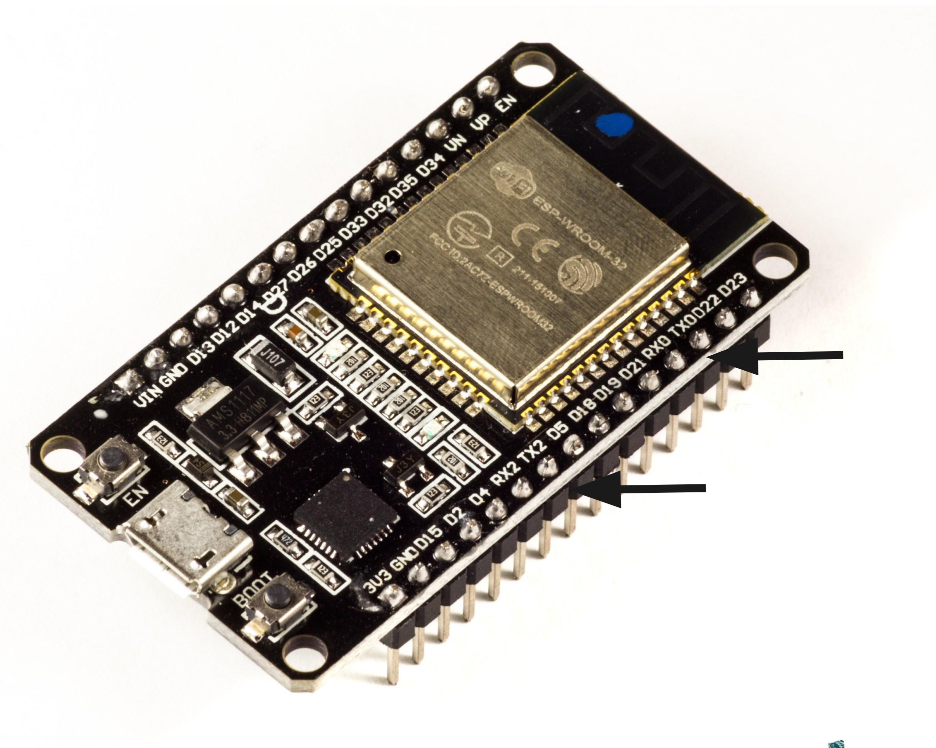
Week 2 | Session 2

UART

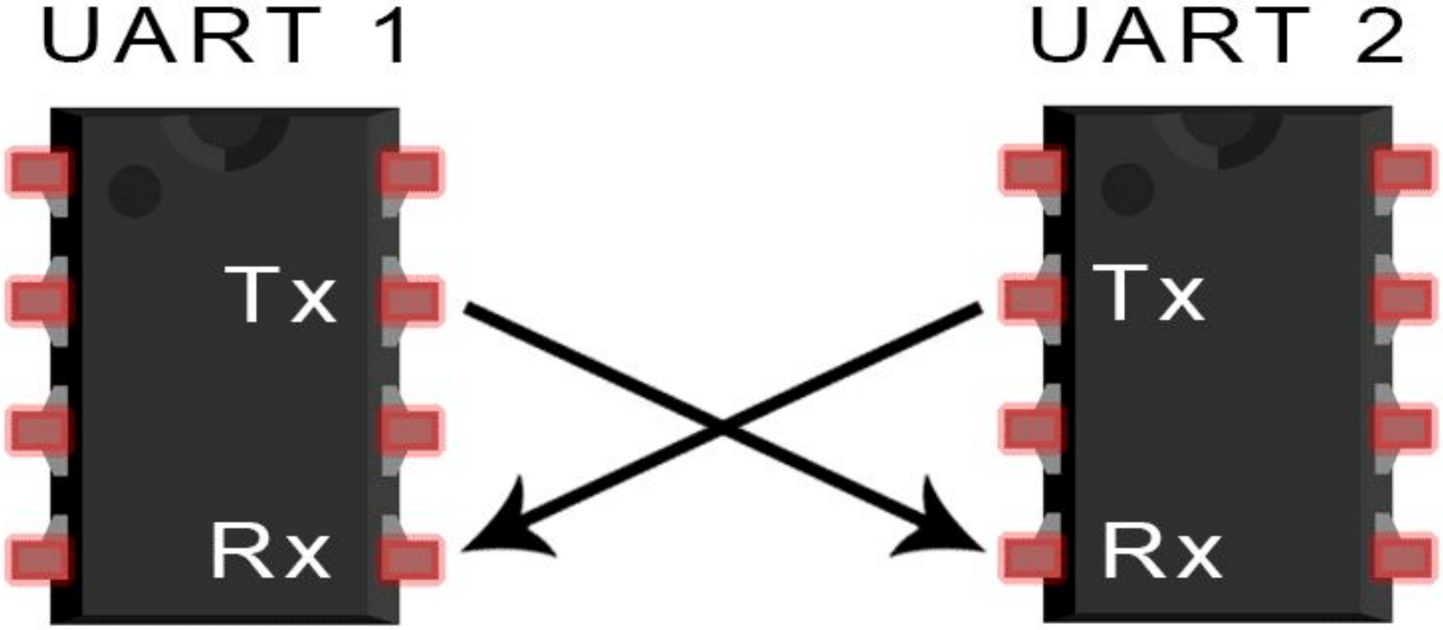
Universal Asynchronous Receiver/Transmitter

UART

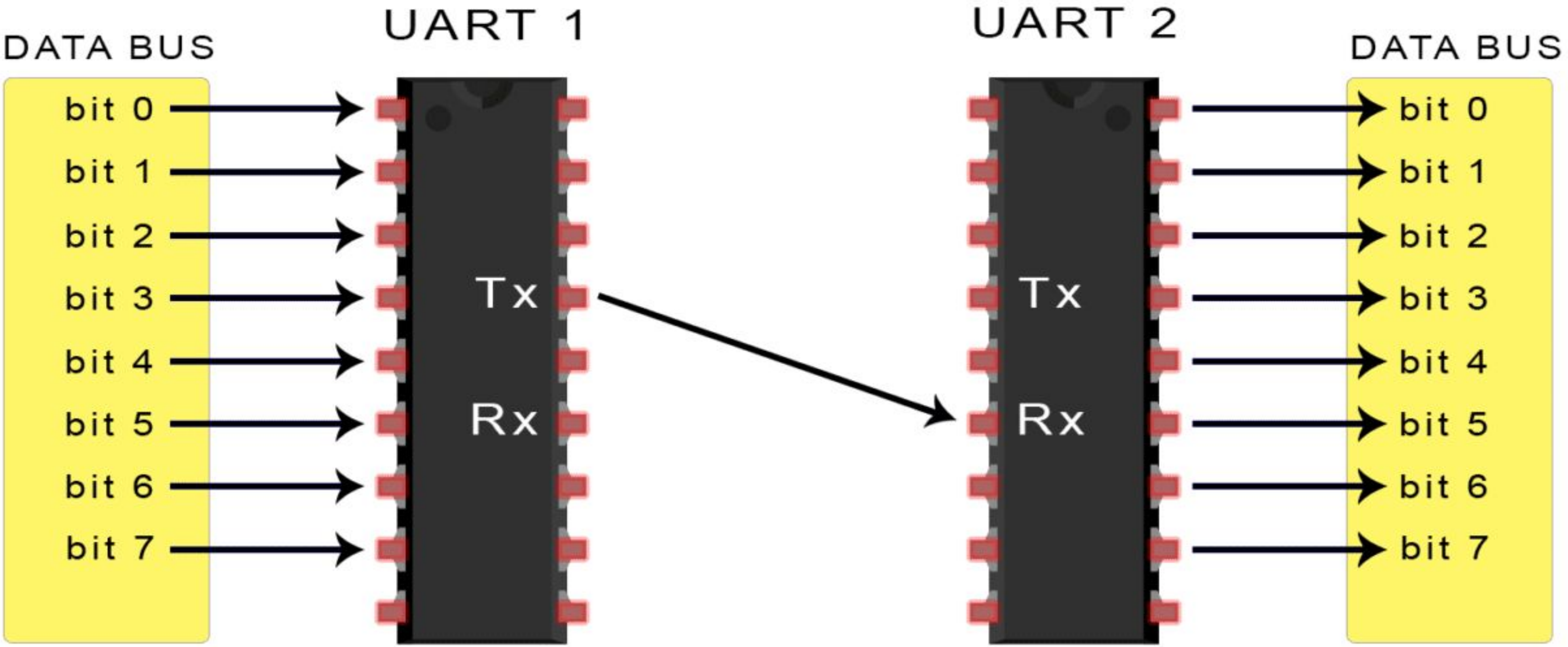

RESET
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
UNO
ARDUINO
ON
ICSP
MADE IN ITALY
ARDUINO.CC
IOREF
RESET
3.3V
5V
GND
Vin
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5


VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
ESP-WROOM-32
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23
EN
BOOT

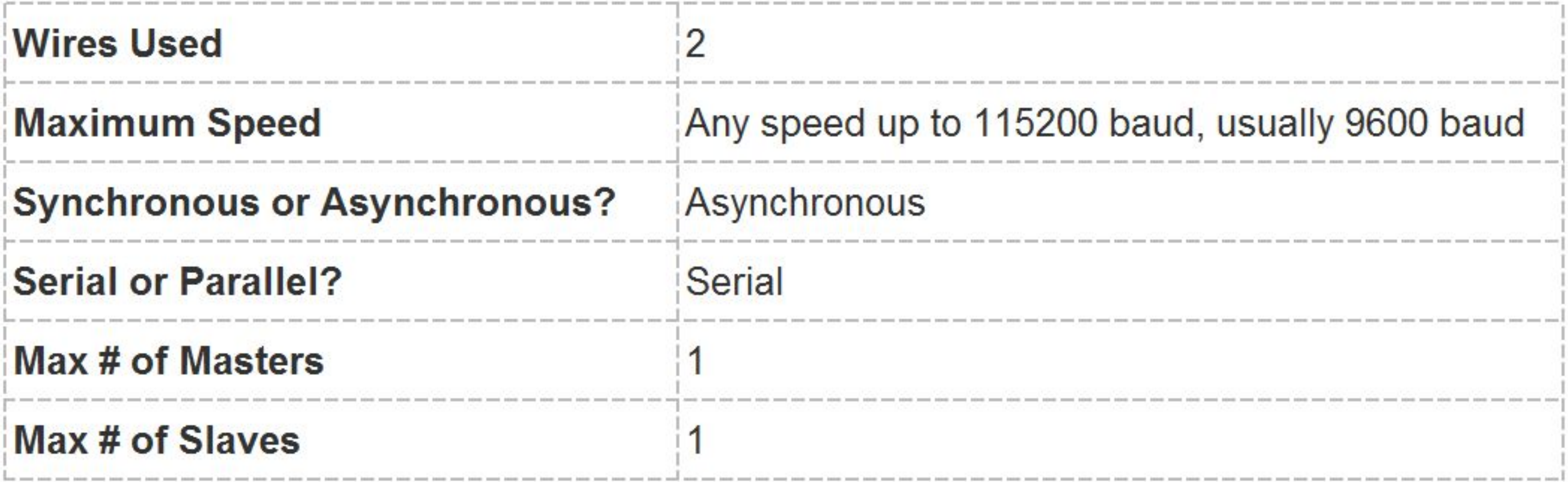
UART



UART

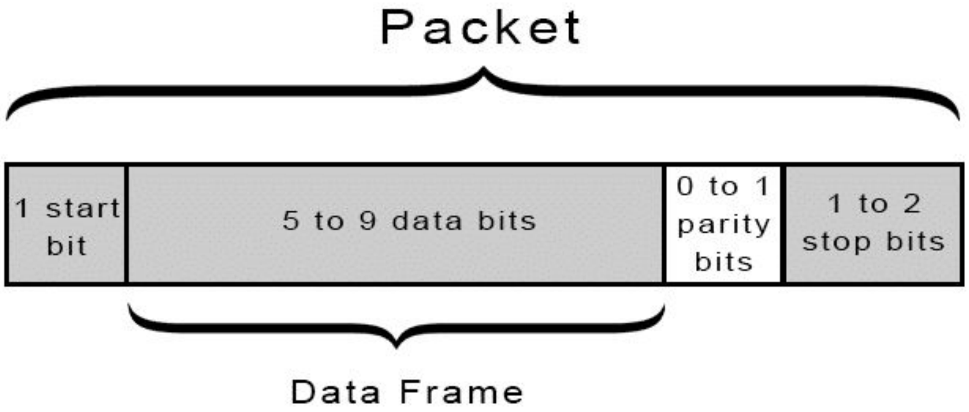

DATA BUS
UART 1
UART 2
DATA BUS
bit 0
bit 1
bit 2
bit 3
bit 4
bit 5
bit 6
bit 7
Tx
Rx
Tx
Rx
bit 0
bit 1
bit 2
bit 3
bit 4
bit 5
bit 6
bit 7

UART



Wires Used	2
Maximum Speed	Any speed up to 115200 baud, usually 9600 baud
Synchronous or Asynchronous?	Asynchronous
Serial or Parallel?	Serial
Max # of Masters	1
Max # of Slaves	1

UART



Packet

1 start bit

5 to 9 data bits

0 to 1 parity bits

1 to 2 stop bits

Data Frame

START BIT

The UART data transmission line is normally held at a high voltage level when it’s not transmitting data. To start the transfer of data, the transmitting UART pulls the transmission line from high to low for one clock cycle. When the receiving UART detects the high to low voltage transition, it begins reading the bits in the data frame at the frequency of the baud rate.

DATA FRAME

The data frame contains the actual data being transferred. It can be 5 bits up to 8 bits long if a parity bit is used. If no parity bit is used, the data frame can be 9 bits long. In most cases, the data is sent with the least significant bit first.

PARITY

Parity describes the evenness or oddness of a number. The parity bit is a way for the receiving UART to tell if any data has changed during transmission. Bits can be changed by electromagnetic radiation, mismatched baud rates, or long distance data transfers. After the receiving UART reads the data frame, it counts the number of bits with a value of 1 and checks if the total is an even or odd number. If the parity bit is a 0 (even parity), the 1 bits in the data frame should total to an even number. If the parity bit is a 1 (odd parity), the 1 bits in the data frame should total to an odd number. When the parity bit matches the data, the UART knows that the transmission was free of errors. But if the parity bit is a 0, and the total is odd; or the parity bit is a 1, and the total is even, the UART knows that bits in the data frame have changed.

STOP BITS

To signal the end of the data packet, the sending UART drives the data transmission line from a low voltage to a high voltage for at least two bit durations.

ADVANTAGES

- Only uses two wires
- No clock signal is necessary
- Has a parity bit to allow for error checking
- The structure of the data packet can be changed as long as both sides are set up for it
- Well documented and widely used method

DISADVANTAGES

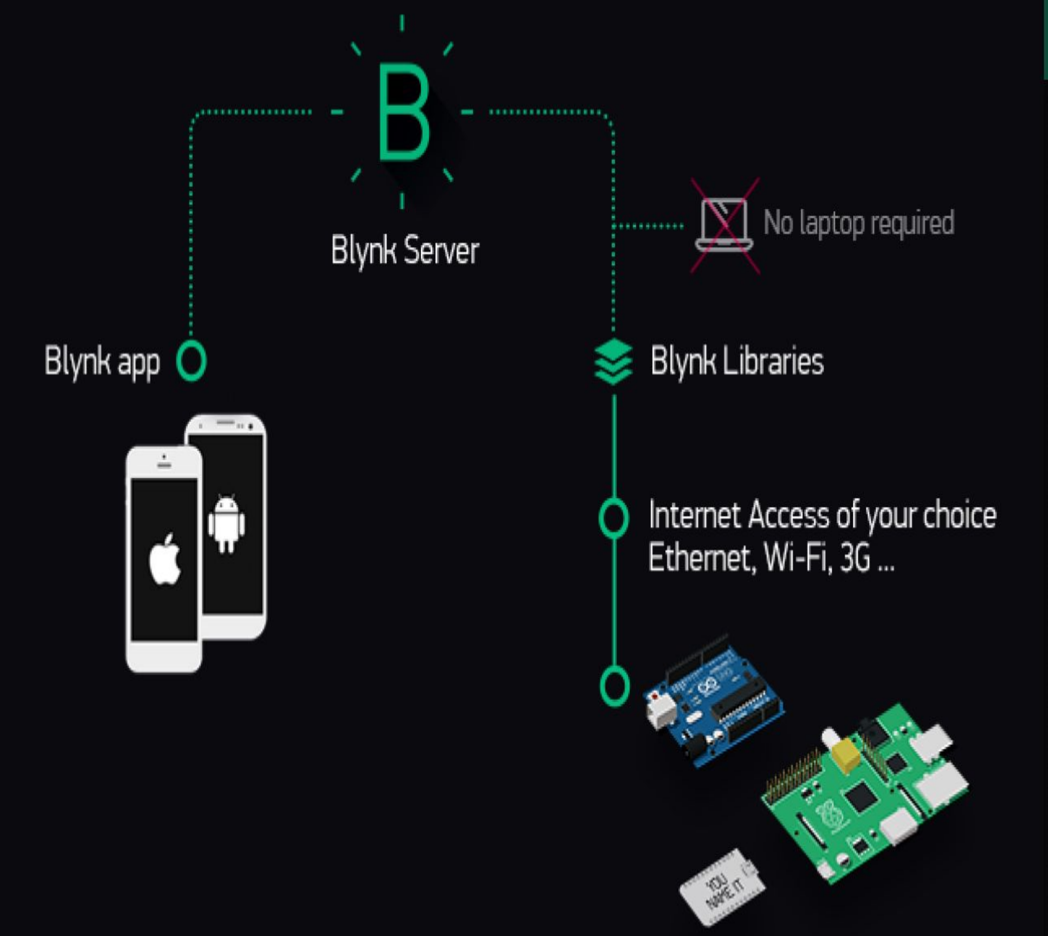
- The size of the data frame is limited to a maximum of 9 bits
- Doesn’t support multiple slave or multiple master systems
- The baud rates of each UART must be within 10% of each other

Blynk App

Getting started with BLYNK

Why BLYNK?

- It’s one of the best known app for doing IoT with any board that you want.
- You can do a lot of fun stuff as we will see further in the session
- It helps you to control hardware via your mobile, displays data of sensors to you.



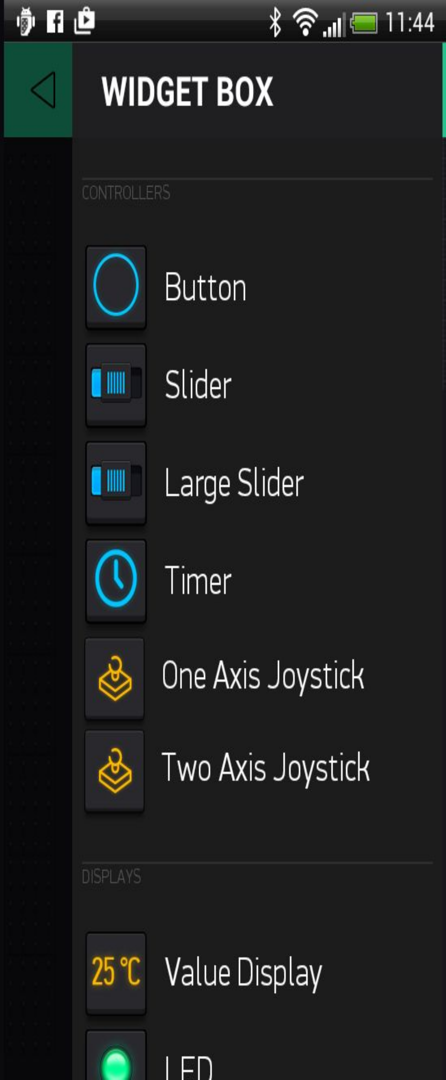
Blynk Server

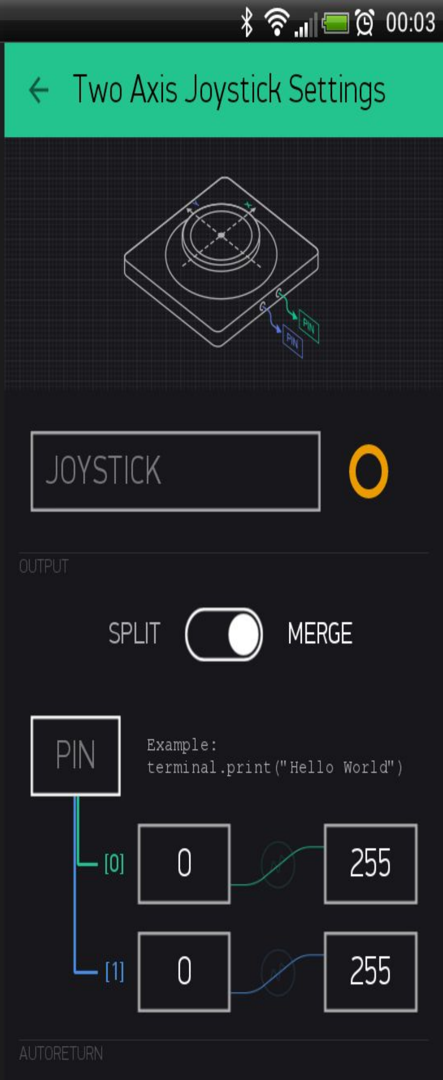
No laptop required

Blynk app

Blynk Libraries

Internet Access of your choice
Ethernet, Wi-Fi, 3G ...





Two Axis Joystick Settings

JOYSTICK

OUTPUT

SPLIT ● MERGE

PIN

```
Example:
terminal.print("Hello World")
```

[0]	0	255
[1]	0	255

AUTORETURN

LIVE DEMO

Virtual Pins

What are they?

- They are some virtual pins exist in the blynk app to send and receive signals with your ESP32 board

Why to use?

- You can use them to control output to real pins according to the code in which input is from virtual pins
- Some frequently used functions in Blynk don’t have real pins.

Functions in Arduino IDE

- BLYNK_WRITE()
- BLYNK_READ()
- Timer

These are some basic ones, there are many more widgets and functions to look into like-

- Bridge
- Joystick and so on

You can try this by yourself for fun

LIVE DEMO

Don’t worry we will be continuing from here in the next session where we will look upon rest of the functions and will also integrate Blynk app with voice assistant and let you make your childhood dream come true.

Other References-

- Basics of UART
- Getting Started with Blynk
- Glowing a Led
- Virtual Pins
- ESP32 Datasheet

Codes we used-

- Example Code for all the widgets and functions
- Chatting b/w terminal and serial monitor