Actividad 1\_ Código

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <BluetoothSerial.h>

BluetoothSerial SerialBT;

LiquidCrystal\_I2C lcd(0x27, 16, 2); // Dirección I2C del LCD (puede ser 0x3F en algunos módulos)

void setup() {

Serial.begin(115200);

SerialBT.begin("ESP32\_LCD"); // Nombre Bluetooth

lcd.init();

lcd.backlight();

lcd.setCursor(0, 0);

lcd.print("Esperando BT...");

}

void loop() {

if (SerialBT.available()) {

String mensaje = SerialBT.readStringUntil('\n'); // Espera mensaje terminado en nueva línea

lcd.clear();

lcd.setCursor(0, 0);

// Si el mensaje es muy largo, dividir en dos líneas

if (mensaje.length() <= 16) {

lcd.print(mensaje);

} else {

lcd.print(mensaje.substring(0, 16));

lcd.setCursor(0, 1);

lcd.print(mensaje.substring(16, 32)); // 2da línea hasta 16 más

}

Serial.println("Mensaje recibido: " + mensaje);

}

}  
  
  
Actividad 2\_Código

#include <ESP32Servo.h>

#include "BluetoothSerial.h"

Servo servo;

BluetoothSerial BT;

const int SERVO\_PIN = 2; // pin GPIO2 para el servo

String inbuf;

void setup() {

// Inicializa servo

ESP32PWM::allocateTimer(0);

ESP32PWM::allocateTimer(1);

ESP32PWM::allocateTimer(2);

ESP32PWM::allocateTimer(3);

servo.setPeriodHertz(50);

servo.attach(SERVO\_PIN, 500, 2400);

servo.write(90); // posición inicial al centro

// Inicializa Bluetooth

BT.begin("ESP32\_SERVO"); // Nombre para emparejar desde el celular

// BT.setPin("1234"); // Opcional: PIN de emparejamiento

}

void loop() {

while (BT.available()) {

char c = (char)BT.read();

if (c == '\r') continue; // Ignora retorno de carro

if (c == '\n') { // Fin de línea = comando completo

int angulo = inbuf.toInt(); // convierte a número

angulo = constrain(angulo, 0, 180); // limita entre 0 y 180

servo.write(angulo); // mueve el servo

BT.printf("Servo movido a %d grados\n", angulo);

inbuf = "";

} else {

inbuf += c;

if (inbuf.length() > 5) inbuf = ""; // seguridad

}

}

}

Codigo final de la 2da activdad