



# PROYECTO: CONTROL POR VOZ Y BLUETOOTH CON ESP32

AUTOMATIZACIÓN DE LUCES Y MONITOREO CON SENSOR LDR

# OBJETIVOS

- - Comunicación Bluetooth entre celular y ESP32
- - Control de 2 LEDs
- - Lectura y visualización del sensor LDR
- - Control por comandos de voz

# COMPONENTES UTILIZADOS

- - ESP32 DevKit
- - Sensor LDR + resistencia
- - 2 LEDs + resistencias
- - Teléfono Android
- - App Inventor
- - Arduino IDE

# FUNCIONAMIENTO DEL SISTEMA

- - ESP32 recibe comandos desde el celular
- - Control de LEDs según comando
- - Envío periódico del valor del sensor
- - App visualiza datos y envía órdenes

# INTERFAZ DE VOZ

- - Uso del componente SpeechRecognizer
- - Comandos simples: encender/apagar luz uno/dos
- - Traducción de voz a texto y envío por Bluetooth



# DISEÑO DE APP (MIT APP INVENTOR)



# PROGRAMA EN BLOQUES PARA LA APP DASHBOARD

initialize global **mensaje\_LDR** to " "

when **Screen1** .Initialize

do set **BluetoothClient1** .DelimiterByte to 10  
set **LDR\_Lbl** .Text to get global **mensaje\_LDR**

when **ListPicker1** .BeforePicking

do set **ListPicker1** .Elements to **BluetoothClient1** .AddressesAndNames

when **ListPicker1** .AfterPicking

do evaluate but ignore result call **BluetoothClient1** .Connect  
address **ListPicker1** .Selection  
set **Label11** .Text to **ListPicker1** .Selection

when **PATIO\_ON** .Click

do call **BluetoothClient1** .SendText  
text "encender lámpara patio\n"

when **PATIO\_OFF** .Click

do call **BluetoothClient1** .SendText  
text "apagar lámpara patio\n"

when **FRENTE\_ON** .Click

do call **BluetoothClient1** .SendText  
text "encender lámpara frente\n"

when **FRENTE\_OFF** .Click

do call **BluetoothClient1** .SendText  
text "apagar lámpara frente\n"

when **VOICE\_COM** .Click

do call **SpeechRecognizer1** .GetText

when **SpeechRecognizer1** .AfterGettingText

result partial  
do call **BluetoothClient1** .SendText text get result

when **Clock1** .Timer

do call **Tomar\_Dato**

to **Tomar\_Dato**

do call **BluetoothClient1** .SendText  
text "informar intensidad de luz"  
if call **BluetoothClient1** .BytesAvailableToReceive > 0  
then set **LDR\_Lbl** .Text to call **BluetoothClient1** .ReceiveText  
numberOfBytes -1

# CÓDIGO ESP32 (RESUMEN)

- - BluetoothSerial para comunicación
- - Comandos para LEDs
- - Lectura de LDR y envío cada 2 segundos

```
void loop() {  
  if (controladorBT.available() > 0){  
    String mensaje = controladorBT.readStringUntil('\n'); //Lee el strig proveniente del t  
    mensaje.trim(); //recorta espacios y saltos  
    Serial.println("Comando Recibido");  
    Serial.println(mensaje);  
  
    if (mensaje.equalsIgnoreCase("encender lámpara patio")){  
      digitalWrite(L_PATIO, HIGH);  
      Serial.println("Lámpara patio encendida");  
    }  
    else if (mensaje.equalsIgnoreCase("apagar lámpara patio")){  
      digitalWrite(L_PATIO, LOW);  
      Serial.println("Lámpara patio apagada");  
    }  
    if (mensaje.equalsIgnoreCase("encender lámpara frente")){  
      digitalWrite(L_FRENTE, HIGH);  
      Serial.println("Lámpara frente encendida");  
    }  
    else if (mensaje.equalsIgnoreCase("apagar lámpara frente")){  
      digitalWrite(L_FRENTE, LOW);  
      Serial.println("Lámpara frente apagada");  
    }  
    else if (mensaje.equalsIgnoreCase("informar intensidad de luz")){  
      int LDR = map(analogRead(34), 0, 4096, 0, 100);  
      Serial.print("La intensidad luminica es de: ");  
      Serial.print(LDR);  
      Serial.println("%");  
      controladorBT.println("La intensidad luminica es de: " + String(LDR)+ "%");  
    }  
    else {  
      Serial.println("Comando incorrecto");  
    }  
  }  
  
  if (millis() - ultimoEnvio > intervalo_lectura){  
    int LDR = map(analogRead(34), 0, 4095, 0, 100);  
    String luz = "La intensidad luminica es de: " + String(LDR) + "%\n";  
    controladorBT.println(luz);  
    Serial.println("Enviado a app: " + luz);  
    ultimoEnvio = millis();  
  }  
}
```



# RESULTADOS

- - Comunicación estable
- - Ejecución correcta de comandos
- - Lectura del LDR visible en app
- - Reconocimiento de voz funcional

# CONCLUSIONES

- - Proyecto exitoso de integración HW+SW
- - Base para domótica por voz
- - Potencial para expandir a control por WiFi/MQTT

## POSIBLES MEJORAS

- - Conexión WiFi
- - Almacenamiento en la nube
- - Interfaz gráfica más amigable
- - Reconocimiento de voz mejorado