

TRABAJO DE FINAL DE GRADO

Grado en Ingeniería electrónica industrial y automática

SISTEMA DE GESTIÓN DOMÓTICA DE UNA VIVIENDA PARA PERSONAS QUE PRESENTAN ALGÚN TIPO DE DISCAPACIDAD



Volumen I

Anexos

Autor: Esther Tadeo Sánchez

Director: Manuel Andrés Manzanares Brotons

Departamento: EEL

Convocatoria: Mayo 2022

Índice de contenido

1. Código IDE Arduino.....	4
1.1 Declaración de librerías	4
1.2 Declaración de variables.....	4
1.3 Declaración de funciones.....	7
1.4 Set Up.....	9
1.5 Loop	11
1.6 Funciones.....	16
2. Código Mit App Inventor	29
2.1 Selección bluetooth.....	29
2.2 Recepción de datos a la aplicación por bluetooth.....	29
2.3 Envío de datos temperatura al microcontrolador	33
2.4 Envío de datos luces al microcontrolador	34
2.5 Envío de datos persianas y puertas al microcontrolador	34
2.6 Envío de datos alarmas.....	34
2.7 Envío de datos frecuencia cardíaca.....	35
2.8 Envío de datos riego de plantas.....	35
2.9 Animación pantalla inicio	35
2.10 Animación pantalla temperatura.....	36
2.11 Animación pantalla iluminación	37
2.12 Animación pantalla persianas.....	37
2.13 Animación pantalla puerta.....	37

2.14 Animación pantalla ritmo cardíaco.....	38
2.15 Animación pantalla riego plantas.....	38
2.16 Animación pantalla alarmas.....	38
2.17 Animación pantalla emergencia.....	39

1. Código IDE Arduino

1.1 Declaración de librerías

```
*****LIBRERIAS*****
//TEMPERATURA
#include <DHT.h>
//MOVIMIENTO
#include <TimerOne.h>
//LECTOR RFID Y MOTOR PASO A PASO (PUERTA)
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>
//LCD
#include <LiquidCrystal.h>
```

1.2 Declaración de variables

```
*****VARIABLES*****
String cadena = {};//Contiene los datos que se imprimen por bluetooth
char dato = 'z'; //Contiene los datos que se reciben por bluetooth
//Guarda el char dato
char dato_temperatura = '1';
char dato_luces = '7';
char dato_riego = 'p';

//TEMPERATURA Y HUMEDAD
float T; //Lee la temperatura
char Switch_temperatura;
#define DHTPIN 4
#define Aire_acondicionado 5 //Led aire acondicionado
#define Calefaccion 6 //Led calefacción
```

```
int ac = 0; //Envia el estado del aire i la calefacción
#define DHTTYPE DHT11
//Configuración del sensor utilizado
DHT dht(DHTPIN,DHTTYPE);

//SENSOR MOVIMIENTO
#define PIRPIN 2
int Time_M = 0;

//LUMINOSIDAD
#define Luz1 7
#define Luz2 8
#define Luz3 9
int Luminosidad; //Mide la luminosidad del pin LDR
#define LDR A0
int Porcentaje_Luminosidad; //Convierte la luminosidad en porcentaje
int Luces = 0; //Envia el estado de las luces

//PERSIANAS
#define E1 10 //Enable 1 convertidor
#define C2 11 //Control pin 2
#define C1 12 //Control pin 1
int Contador_Persiana = 0;

//PUERTA
#define RST_PIN 13 // Reset del RC522
#define SS_PIN 53 //SS (SDA) del RC522
MFRC522 mfrc522(SS_PIN, RST_PIN); //Crear instancia del MFRC522
byte clave_valida[4] = { 0x2A, 0x2E, 0xA8, 0x16 }; // Ejemplo de clave valida
Servo servoMotor;
int Estado_Tarjeta = 0;
int Estado_Puerta = 0;

//SENSOR DE PULSO
#define Pulso A1
#define Pulsador_Pulso 18 //INTERRUPCIÓN
#define Led_Pulso 14
```

```
int Contador_Pulso = 0; //Frecuencia cardíaca
int Timer_Pulso= 0; //Mide 15 segundos
int P; //Longitud de onda

//SENSOR DE GAS
#define Buzzer 15
#define G 19 //Interrupción sensor de gas
#define Pulsador_Stop_Buzzer 20 //Interrupción parar el buzzer
int Gas = 1;

//SENSOR DE LLAMA
#define F 21
#define Buzzer 15
int Fuego = 0;

//SENSOR DE RIEGO
#define H A2 //Sensor de humedad
#define AirValue 760 //Valor sin agua
#define WaterValue 350 //Valor con agua
#define R 22 //Relé bomba
int Humedad = 0;
int Porcentaje_Humedad = 0;

//LCD
LiquidCrystal lcd(28,29,24,25,26,27);
unsigned long PreviousMillis = 0; //Se guarda la última actualización
const long Interval = 1000; //Intervalo al que cambia la pantalla LCD
int Numero_Pantalla = 0; //Cambia dependiendo de la pantalla que se muestre
String Pantalla_Calefaccion; //Indica el estado de la calefacción
String Pantalla_Aire_Acondicionado; //Indica el estado del aire acondicionado
String Pantalla_Luces_1; //Indica el estado de la luz 1
String Pantalla_Luces_2; //Indica el estado de la luz 2
String Pantalla_Luces_3; //Indica el estado de la luz 3
String Pantalla_Gas; //Indica el estado de la alarma de gas
String Pantalla_Fuego; //Indica el estado de la alarma de fuego
```

1.3 Declaración de funciones

```
*****DECLARACIÓN FUNCIONES*****  
//Envia datos por el puerto serie  
void serial_print();  
  
//SENSOR TEMPERATURA  
/*Mide y envia la temperatura periódicamente.*/  
void temperatura();  
/*Comprueba la temperatura periódicamente, si hace frio se enciende la calefacción y si hace calor el aire  
acondicionado*/  
void auto_temperatura();  
/*Enciende el aire acondicionado*/  
void aire_acondicionado();  
/*Enciende la calefacción*/  
void calefaccion();  
  
//SENSOR MOVIMIENTO  
/* se encienden las luces en función de la luminosidad y se activa un temporizador (Timer.1) de 1 segundo.  
 * Si se vuelve a detectar presencia se reinicia. Se reinicia la variable Time_M a 0 segundos*/  
void encender_luces();  
/* Se apagan las luces y se desactiva el Timer 1.*/  
void apagar_luces();  
/*Mide el porcentaje de luminosidad*/  
void luminosidad();  
/*Cuando no hay suficiente luminosidad activa la interrupción del PIN 2 para encender_luces().  
 * Cuando hay suficiente luminosidad desactiva la interrupción del PIN 2 y se llama a apagar_luces()*/  
void auto_luminosidad();  
/* Preescalor que cuenta 5 veces 1 segundo. Cuando desborda se llama a apagar_luces().*/  
void temporizador();  
  
//PERSIANAS  
/*Enciende el motor para abrir la persiana */  
void abrir_persiana();  
/*Enciende el motor para cerrar la persiana */  
void cerrar_persiana();
```

```
/*Para el motor para detener la persiana */
void parar_persiana();

//PUERTA
/*Comprueba el lector RFID periódicamente, cuando detecta tarjeta valida llama a las funciones
 * abrir o cerrar puerta dependiendo el estado de la puerta */
void lector_rfid();
/*Activa el motor paso a paso para abrir la puerta*/
void abrir_puerta();
/*Activa el motor paso a paso para cerrar la puerta*/
void cerrar_puerta();
/*Función para comparar dos vectores. Comparamos el lector con la tarjeta*/
bool isEqualArray(byte* arrayA, byte* arrayB, int length){
    for (int index = 0; index < length; index++){
        if (arrayA[index] != arrayB[index]) return false;
    }
    return true;
}

//SENSOR PULSO
/*Mediante un pulsador conectado a la interrupción 18 se mide el pulso durante 15 segundos*/
void sensor_pulso();

//SENSOR GAS
/*Lee el bit del sensor de gas*/
void sensor_gas();

//SENSOR LLAMA
/*Lee el bit del sensor de llama*/
void sensor_llama();

//BUZZER ALARMA
/*Hace sonar la alarma*/
void buzzer();
// SRTOP BUZZER ALARMA
/*Detiene la alarma*/
void stop_buzzer();
```

```
//RIEGO PLANTAS
/*Mide el porcentaje de humedad en las plantas*/
void sensor_humedad();
/*Reiga las plantas manualmente*/
void riego_plantas_on();
/*Para el reiego de plantas manualmente*/
void riego_plantas_off();
/*Reiga las plantas*/
void auto_riego_plantas();

//Pantalla LCD
void pantalla_lcd();
```

1.4 Set Up

```
*****SETUP*****
void setup() {
    //Inicializar puerto serie a 9600 baudios
    Serial.begin(9600);
    //Se inicializa el sensor DHT11
    dht.begin();
    //Se inicia el bus SPI
    SPI.begin();
    //Se inicia el lector RFID
    mfrc522.PCD_Init();
    //Se inicia la pantalla LCD 16x2
    lcd.begin(16, 2);

    //Pins
    pinMode(DHTPIN, INPUT);
    pinMode(Aire_acondicionado, OUTPUT);
    digitalWrite(Aire_acondicionado, LOW);
    pinMode(Calefaccion, OUTPUT);
    digitalWrite(Calefaccion, LOW);
```

```
pinMode(PIRPIN, INPUT);
pinMode(Luz1, OUTPUT);
digitalWrite(Luz1, LOW);
pinMode(Luz2, OUTPUT);
digitalWrite(Luz2, LOW);
pinMode(Luz3, OUTPUT);
digitalWrite(Luz3, LOW);
pinMode( E1, OUTPUT);
digitalWrite( E1, LOW);
pinMode( C1, OUTPUT);
digitalWrite( C1, LOW);
pinMode( C2, OUTPUT);
digitalWrite( C2, LOW);
servoMotor.attach(3); //Pin 3 motor paso a paso
pinMode(Pulso, INPUT);
pinMode(Pulsador_Pulso, INPUT);
pinMode(Led_Pulso, OUTPUT);
digitalWrite(Led_Pulso, LOW);
//Cuando se pulsa cambia de '0' --> '1' y mide el pulso
attachInterrupt(digitalPinToInterrupt(Pulsador_Pulso), sensor_pulso, RISING);
pinMode(G, INPUT);
//Alarma de gas. Cuando cambia de '1' --> '0' (a 406 ppm) suena el buzzer
attachInterrupt(digitalPinToInterrupt(G), buzzer , FALLING);
Pantalla_Gas = "Activada";
pinMode(F, INPUT);
//Alarma de incendio. Cuando cambia de '0' --> '1' suena el buzzer
attachInterrupt(digitalPinToInterrupt(F), buzzer , RISING);
Pantalla_Fuego = "Activada";
pinMode(Pulsador_Stop_Buzzer, INPUT);
//Cuando cambia de '1' --> '0' para el buzzer
attachInterrupt(digitalPinToInterrupt(Pulsador_Stop_Buzzer), stop_buzzer , FALLING);
pinMode(H, INPUT);
pinMode(R, OUTPUT);
digitalWrite(R, HIGH);
}
```

1.5 Loop

```
*****LOOP*****
void loop() {
    if (Serial.available()) {
        dato = Serial.read();
    }
    //-----TEMPERATURA-----
    //Temperatura manual mediante el Switch
    if (dato == '0'){
        //Desactivamos la temperatura automática
        dato_temperatura = 0;
        ac = 0; //Ninguna acción asociada
    }
    //Calefacción y aire acondicionado automático mediante el Switch
    else if (dato == '1'){
        //Guardamos el '1'
        dato_temperatura = dato;
    }
    //Encender calefacción manual
    else if (dato == '2'){
        calefaccion();
    }
    //Apagar calefacción manual
    else if (dato == '3'){
        digitalWrite(Calefaccion,LOW);
        Pantalla_Calefaccion = "OFF";
    }
    //Encender aire acondicionado manual
    else if (dato == '4'){
        aire_acondicionado();
    }
    //Apagar aire acondicionado manual
    else if (dato == '5'){
        digitalWrite(Aire_acondicionado,LOW);
        Pantalla_Aire_Acondicionado = "OFF";
    }
}
```

```
}

//-----LUCES-----
//Luces manual mediante el Switch
else if (dato == '6') {
    detachInterrupt(digitalPinToInterrupt(PIRPIN));
    //Desactivamos las luces automáticas
    dato_luces = 0;
    Luces = 0; //Ninguna acción asociada
}
//Luces automáticas mediante el Switch
else if (dato == '7') {
    //Guardamos el '7'
    dato_luces = dato;
}
//Encender Luz 1
else if (dato == '8') {
    digitalWrite(Luz1,HIGH);
    Pantalla_Luces_1 = " ON";
}
//Apagar Luz 1
else if (dato == '9') {
    digitalWrite(Luz1,LOW);
    Pantalla_Luces_1 = "OFF";
}
//Encender Luz 2
else if (dato == 'a') {
    digitalWrite(Luz2,HIGH);
    Pantalla_Luces_2 = " ON";
}
//Apagar Luz 2
else if (dato == 'b') {
    digitalWrite(Luz2,LOW);
    Pantalla_Luces_2 = "OFF";
}
//Encender Luz 3
else if (dato == 'c') {
    digitalWrite(Luz3,HIGH);
```

```

Pantalla_Luces_3 = " ON";
}
//Apagar Luz 3
else if (dato == 'd') {
    digitalWrite(Luz3,LOW);
    Pantalla_Luces_3 = "OFF";
}
//-----PERSIANAS-----
//Cerrar persiana
else if (dato == 'e') {
    cerrar_persiana();
}
//Parar persiana
else if (dato == 'f') {
    parar_persiana();
}
//Abrir persiana
else if (dato == 'g') {
    abrir_persiana();
}
//-----PUERTAS-----
else if (dato == 'h') {
    abrir_puerta();
    dato = 'Z';
}
else if (dato == 'i') {
    cerrar_puerta();
    dato = 'Z';
}

//-----ALARMAS-----
//Alarma gas automática
else if (dato == 'j') {
    //Cuando cambia de '1' --> '0' suena el buzzer
    attachInterrupt(digitalPinToInterrupt(G), buzzer , FALLING);
    Pantalla_Gas = "Activada";
}

```

```

//Alarma gas desactivada
else if (dato == 'k'){
    detachInterrupt(digitalPinoToInterrupt(G));
    Pantalla_Gas = "Desactivada";
}
//Alarma incendio automática
else if (dato == 'l'){
    //Cuando cambia de '0' --> '1' suena el buzzer
    attachInterrupt(digitalPinoToInterrupt(F), buzzer , RISING);
    Pantalla_Fuego = "Activada";
    stop_buzzer();
}
//Alarmas incendio desactivada
else if (dato == 'm'){
    detachInterrupt(digitalPinoToInterrupt(F));
    Pantalla_Fuego = "Desactivada";
}
//Parar alarma mediante pulsador
else if (dato == 'n'){
    stop_buzzer();
}

//-----RITMO CARDIACO-----
else if (dato == 'o'){
    sensor_pulso();
    dato = 'z';
}

//-----RIEGO PLANTAS-----
//Riego manual mediante el Switch
if (dato == 'q'){
    //Desactivamos el riego automático
    dato_rieo = 0;
    dato = 'z'; //Creo que no hace falta
}
//Riego automático mediante el Switch
else if (dato == 'p'){

```

```
//Guardamos la 'p'  
dato_riego = dato;  
}  
//Encender riego  
else if (dato == 'r') {  
    riego_plantas_on();  
}  
//Apagar riego  
else if (dato == 's') {  
    riego_plantas_off();  
}  
  
//-----INICIACION-----  
else if (dato == 'z') {  
    //stop_buzzer();  
}  
  
//-----SWITCH MANUAL-----  
//Temperatura automática  
if (dato_temperatura == '1') {  
    auto_temperatura();  
}  
//Luces automáticas  
if (dato_luces == '7') {  
    auto_luminosidad();  
}  
//Riego automático  
if (dato_riego == 'p') {  
    auto_riego_plantas();  
}  
  
//-----LLAMADA FUNCIONES-----  
//Comprobar temperatura  
temperatura();  
serial_print();  
//Comprobar la luminosidad
```

```

luminosidad();
serial_print();
//Comprobar lector RFID puerta
lector_rfid();
//Comprobar presencia de gas
sensor_gas();
serial_print();
//Comprobar presencia de llama
sensor_llama();
serial_print();
//Comprueba el porcentaje de humedad en las plantas
sensor_humedad();
serial_print();
//Muestra los valores en la pantalla LCD
pantalla_lcd();
}

```

1.6 Funciones

```

/*********************FUNCIONES****************************/
//-----TEMPERATURA-----
void temperatura(){
    T = dht.readTemperature();
}

//-----AUTO TEMPERATURA-----
void auto_temperatura(){
    //Si la temperatura no es la adecuada se redirige a la función
    if (T < 19){
        calefaccion();
        ac = 1; //Enciende Led calefacción ON
    }
    else if (T > 27){
        aire_acondicionado();
        ac = 2; //Enciende Led aire acondicionado ON
    }
}

```

```

}

//Si esta entre 17 y 26 grados se apagan la calefacción y el aire
else{
    digitalWrite(Calefaccion,LOW);
    digitalWrite(Aire_acondicionado,LOW);
    ac = 3; //Led aire acondicionado y calefacción OFF
    Pantalla_Calefaccion = "OFF";
    Pantalla_Aire_Acondicionado = "OFF";
}
}

//-----AIRE ACONDICIONADO-----
void aire_acondicionado(){
    digitalWrite(Aire_acondicionado,HIGH);
    //No permite que esten encendidos a la vez
    digitalWrite(Calefaccion,LOW);
    Pantalla_Calefaccion = "OFF";
    Pantalla_Aire_Acondicionado = "ON";
}

//-----CALEFACCIÓN-----
void calefaccion(){
    digitalWrite(Calefaccion,HIGH);
    //No permite que esten encendidos a la vez
    digitalWrite(Aire_acondicionado,LOW);
    Pantalla_Calefaccion = "ON";
    Pantalla_Aire_Acondicionado = "OFF";
}

//-----LUMINOSIDAD-----
void luminosidad(){
    Luminosidad = analogRead(LDR);
    //Creamos el porcentaje de iluminación
    Porcentaje_Luminosidad = map(Luminosidad, 0, 200, 0, 100);
}

//-----AUTO LUMINOSIDAD-----

```

```

void auto_luminosidad(){
    if (Porcentaje_Luminosidad < 90){
        //Cuando no haya suficiente luminosidad se activa la interrupción del PIN 2
        attachInterrupt(digitalPinToInterrupt(PIRPIN), encender_luces, RISING);
    }
    else{
        //Cuando haya suficiente luminosidad se desactiva la interrupción del PIN 2
        detachInterrupt(digitalPinToInterrupt(PIRPIN));
        //Se apagan las luces
        apagar_luces();
    }
}

//-----ENCENDER LUCES-----
void encender_luces(){
    //Se encienden las luces en función de la luminosidad
    //Si es inferior a 40%
    if (Porcentaje_Luminosidad < 40){
        digitalWrite(Luz1,HIGH);
        digitalWrite(Luz2,HIGH);
        digitalWrite(Luz3,HIGH);
        Luces = 1; //LED Luz 1, Luz 2, Luz 3 ON
        Pantalla_Luces_1 = " ON";
        Pantalla_Luces_2 = " ON";
        Pantalla_Luces_3 = "ON";
    }
    //Si está entre 40% y 60%
    else if (Porcentaje_Luminosidad < 60){
        digitalWrite(Luz1,HIGH);
        digitalWrite(Luz2,HIGH);
        digitalWrite(Luz3,LOW);
        Luces = 2; //LED Luz 1, Luz 2, ON. LED Luz 3 OFF
        Pantalla_Luces_1 = " ON";
        Pantalla_Luces_2 = " ON";
        Pantalla_Luces_3 = "OFF";
    }
    //Si está entre 60% y 90%
}

```

```

else if (Porcentaje_Luminosidad < 90){
    digitalWrite(Luz1,HIGH);
    digitalWrite(Luz2,LOW);
    digitalWrite(Luz3,LOW);
    Luces = 3; //LED Luz 1 ON. LED Luz 2, Luz 3 OFF
    Pantalla_Luces_1 = " ON";
    Pantalla_Luces_2 = "OFF";
    Pantalla_Luces_3 = "OFF";
}
//Se inicia el timer1 a 1 segundo y la cuenta a 0
Timer1.initialize(1000000);
Time_M = 0;
//Activa la interrupción del Timer 1 para que desborde en Temporizador
Timer1.attachInterrupt(temporizador);
}

//-----APAGAR LUCES-----
void apagar_luces(){
//Se apagan las luces
    digitalWrite(Luz1,LOW);
    digitalWrite(Luz2,LOW);
    digitalWrite(Luz3,LOW);
    Luces = 4; //LED Luz 1, Luz 2, Luz 3 OFF
    Pantalla_Luces_1 = "OFF";
    Pantalla_Luces_2 = "OFF";
    Pantalla_Luces_3 = "OFF";
//Deshabilitar temporizador
    Timer1.detachInterrupt();
}

//-----TEMPORIZADOR-----
void temporizador(){
    if(Time_M > 15){
        //Resetea el contador cuando llega a 15 segundos
        Time_M=0;
        //Apaga las luces
        apagar_luces();
    }
}

```

```

    }
else{
    //Incrementa el timer
    Time_M++;
}
}

//-----CERRAR PERSIANA-----
void cerrar_persiana(){
if (Contador_Persiana < 1){
    digitalWrite(E1, LOW);
}
else{
    Contador_Persiana = Contador_Persiana - 1;
    digitalWrite(E1, HIGH); //Activar motor
    digitalWrite(C1, LOW); //Giro antihorario
    digitalWrite(C2, HIGH);
}
}

//-----PARAR PERSIANA-----
void parar_persiana(){
    digitalWrite(E1, LOW);
}

//-----ABRIR PERSIANA-----
void abrir_persiana(){
if (Contador_Persiana > 10){
    digitalWrite(E1, LOW);
}
else{
    Contador_Persiana = Contador_Persiana + 1;
    digitalWrite(E1, HIGH); //Activar motor
    digitalWrite(C1, HIGH); //Giro horario
    digitalWrite(C2, LOW);
}
}
}

```

```

//-----LECTOR RFID-----
void lector_rfid(){
    // Si se detecta tarjeta
    if (mfrc522.PICC_IsNewCardPresent())
    {
        //Se selecciona una tarjeta
        if (mfrc522.PICC_ReadCardSerial())
        {
            // Se Compara la ID con las claves válidas. 1 VALIDA 2 NO VALIDA
            if (isEqualArray(mfrc522.uid.uidByte, clave_valida, 4)){
                //Comprueba el estado de la puerta. 0 CERRADA 1 ABIERTA
                if (Estado_Puerta == 0){
                    abrir_puerta();
                }
                else if (Estado_Puerta == 1){
                    cerrar_puerta();
                }
                Estado_Tarjeta = 1; //Tarjeta válida
            }
            else{
                Estado_Tarjeta = 2; //Tarjeta NO válida
                //buzzer(); //Alarma intrusión
            }
            // Finalizar lectura actual
            mfrc522.PICC_HaltA();
        }
    }
}

//-----ABRIR PUERTA-----
void abrir_puerta(){
    if (Estado_Puerta == 0){
        // Desplazamos a la posición 180°
        servoMotor.write(180);
        delay(200);
        Estado_Puerta = 1; //Puerta abierta
    }
}

```

```

    }

}

//-----CERRAR PUERTA-----
void cerrar_puerta(){
    if (Estado_Puerta == 1){
        // Desplazamos a la posición 90°
        servoMotor.write(90);
        delay(200);
        Estado_Puerta = 0; //Puerta cerrada
    }
}

//-----SENSOR PULSO-----
void sensor_pulso(){
    Contador_Pulso = 0;
    Timer_Pulso = 0;
    while (Timer_Pulso < 30000){
        P = analogRead(Pulso);
        if (P > 805){
            digitalWrite(Led_Pulso,HIGH);
            Contador_Pulso = Contador_Pulso + 1;
            //Serial.println(P);
        }
        else{
            digitalWrite(Led_Pulso,LOW);
        }
        Timer_Pulso = Timer_Pulso + 1;
    }
    digitalWrite(Led_Pulso,LOW);
    Contador_Pulso = Contador_Pulso/23*4;
    serial_print();
}

//-----SENSOR GAS-----
void sensor_gas(){
    Gas = digitalRead(G);
}

```

```
}

//-----SENSOR LLAMA-----
void sensor_llama(){
    Fuego = digitalRead(F);
}

//-----BUZZER-----
void buzzer(){
    tone(Buzzer, 600);
    serial_print();
}

//-----STOP BUZZER-----
void stop_buzzer(){
    noTone(Buzzer);
    dato = 'z';
    serial_print();
}

//-----SENSOR HUMEDAD-----
void sensor_humedad(){
    //Mide el valor de humedad en las plantas
    Humedad = analogRead(H);
    //Se interpola para que quede en porcentaje
    Porcentaje_Humedad = map(Humedad, AirValue, WaterValue, 0, 100);
}

//-----AUTO RIEGO-----
void auto_riego_plantas(){
    //Si la humedad es inferior a 50% se riega la planta
    if (Porcentaje_Humedad > 40){
        //El relé se pone a LOW porque es lógica invertida
        digitalWrite(R,HIGH);
    }
    else{
        //El relé se pone a HIGH porque es lógica invertida
        digitalWrite(R,LOW);
    }
}
```

```

    }

}

//-----RIEGO ON-----
void riego_plantas_on(){
    digitalWrite(R,LOW);
}

//-----RIEGO OFF-----
void riego_plantas_off(){
    digitalWrite(R,HIGH);
}
//-----PANTALL LCD-----
void pantalla_lcd(){
    // Se comprueba si la diferencia entre el tiempo transcurrido y la última vez
    // que la pantalla cambió es más grande que el intervalo definido.
    // Cuando se cumple la condición la pantalla cambia.
    unsigned long CurrentMillis = millis();
    if (CurrentMillis - PreviousMillis >= Interval) {
        //Guardamos el tiempo en que la pantalla cambió
        PreviousMillis = CurrentMillis;
        // En función del número de pantalla se cambia ésta
    }
    //-----PANTALLA TEMPERATURA-----
    if (Numero_Pantalla == 0) {
        lcd.clear();
        lcd.setCursor(2, 0);
        lcd.print("Temperatura");
        lcd.setCursor(2, 1);
        lcd.print(T);
        lcd.setCursor(8, 1);
        lcd.print("grados");
        Numero_Pantalla = 1;
    }
    //-----PANTALLA CALEFACCIÓN-----
    else if (Numero_Pantalla == 1) {
        lcd.clear();
        lcd.setCursor(2, 0);

```

```
lcd.print("Calefaccion");
lcd.setCursor(6, 1);
lcd.print(Pantalla_Calefaccion);
Numero_Pantalla = 2;
}
//-----PANTALLA AIRE ACONDICIONADO-----
else if (Numero_Pantalla == 2) {
    lcd.clear();
    lcd.setCursor(4, 0);
    lcd.print("Aire AC");
    lcd.setCursor(6, 1);
    lcd.print(Pantalla_Aire_Acondicionado);
    Numero_Pantalla = 3;
}
//-----PANTALLA LUMINOSIDAD-----
else if (Numero_Pantalla == 3) {
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("Luminosidad");
    lcd.setCursor(6, 1);
    lcd.print(Porcentaje_Luminosidad);
    lcd.setCursor(9, 1);
    lcd.print("%");
    Numero_Pantalla = 4;
}
//-----PANTALLA LUZ 1-----
else if (Numero_Pantalla == 4) {
    lcd.clear();
    lcd.setCursor(5, 0);
    lcd.print("Luz 1");
    lcd.setCursor(6, 1);
    lcd.print(Pantalla_Luces_1);
    Numero_Pantalla = 5;
}
//-----PANTALLA LUZ 2-----
else if (Numero_Pantalla == 5) {
    lcd.clear();
```

```
lcd.setCursor(5, 0);
lcd.print("Luz 2");
lcd.setCursor(6, 1);
lcd.print(Pantalla_Luces_2);
Numero_Pantalla = 6;
}
//-----PANTALLA LUZ 3-----
else if (Numero_Pantalla == 6) {
    lcd.clear();
    lcd.setCursor(5, 0);
    lcd.print("Luz 3");
    lcd.setCursor(6, 1);
    lcd.print(Pantalla_Luces_3);
    Numero_Pantalla = 7;
}
//-----PANTALLA PERSIANA-----
else if (Numero_Pantalla == 7) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Estado Persiana");
    if (Contador_Persiana < 1){
        lcd.setCursor(4, 1);
        lcd.print("Cerrada");
    }
    else if (Contador_Persiana < 9){
        lcd.setCursor(0, 1);
        lcd.print("Entre abierta");
    }
    else if (Contador_Persiana < 12){
        lcd.setCursor(5, 1);
        lcd.print("Abierta");
    }
    Numero_Pantalla = 8;
}
//-----PANTALLA TARJETA-----
else if (Numero_Pantalla == 8) {
    lcd.clear();
```

```
lcd.setCursor(1, 0);
lcd.print("Estado Tarjeta");
if (Estado_Tarjeta == 1){
    lcd.setCursor(4, 1);
    lcd.print("Valida");
}
else if (Estado_Tarjeta == 2){
    lcd.setCursor(3, 1);
    lcd.print("No valida");
}
Numero_Pantalla = 9;
}
//-----PANTALLA PUERTA-----
else if (Numero_Pantalla == 9) {
    lcd.clear();
    lcd.setCursor(1, 0);
    lcd.print("Estado Puerta");
    if (Estado_Puerta == 0){
        lcd.setCursor(4, 1);
        lcd.print("Cerrada");
    }
    else if (Estado_Puerta == 1){
        lcd.setCursor(4, 1);
        lcd.print("Abierta");
    }
    Numero_Pantalla = 10;
}
//-----PANTALLA ALARMA GAS-----
else if (Numero_Pantalla == 10) {
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("Alarma Gas");
    lcd.setCursor(3, 1);
    lcd.print(Pantalla_Gas);
    Numero_Pantalla = 11;
}
//-----PANTALLA ALARMA FUEGO-----
```

```

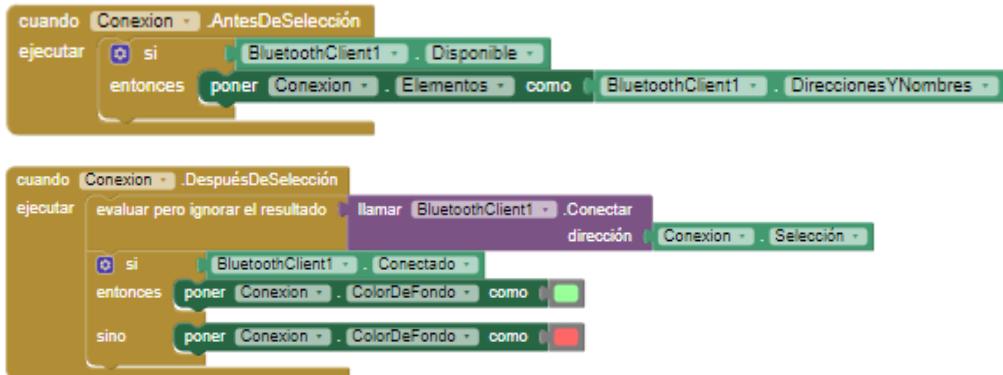
else if (Numero_Pantalla == 11) {
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("Alarma Fuego");
    lcd.setCursor(3, 1);
    lcd.print(Pantalla_Fuego);
    Numero_Pantalla = 12;
}
//-----PANTALLA RIEGO-----
else if (Numero_Pantalla == 12) {
    lcd.clear();
    lcd.setCursor(3, 0);
    lcd.print("% humedad");
    lcd.setCursor(6, 1);
    lcd.print(Porcentaje_Humedad);
    lcd.setCursor(9, 1);
    lcd.print("%");
    Numero_Pantalla = 0;
}
}

//-----SERIAL PRINT-----
void serial_print(){
    cadena = String(T) + " °C" + ", " + ac + " , " + Porcentaje_Luminosidad + " %" + " , " + Luces + " , " +
    Contador_Persiana + " , " + Estado_Tarjeta + " , " + Estado_Puerta + " , " + Contador_Pulso + " , " + Gas + " , " +
    Fuego + " , " + Porcentaje_Humedad + " , " + "%" + " , " + dato;
    Serial.println(cadena);
    delay(300);
}

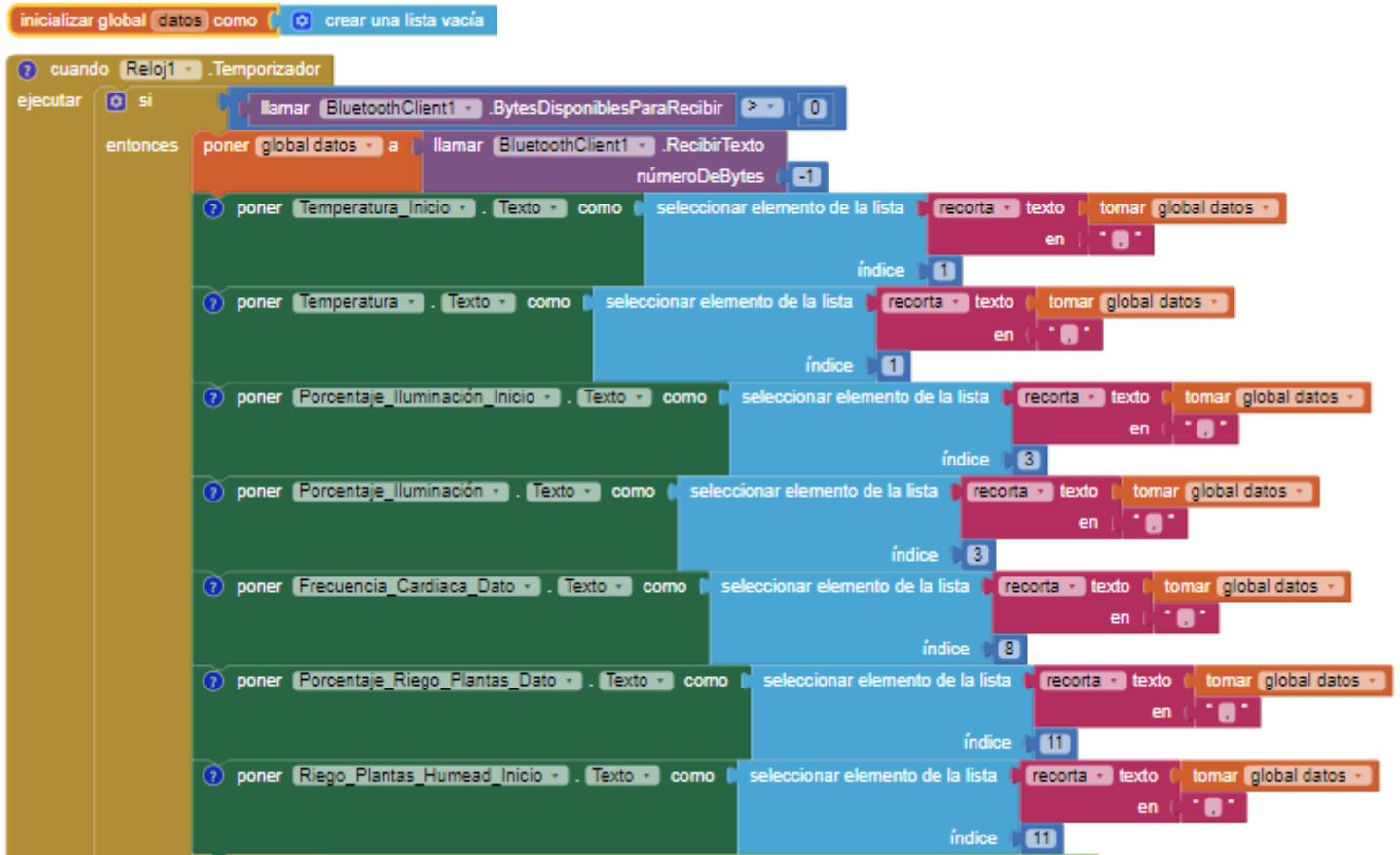
```

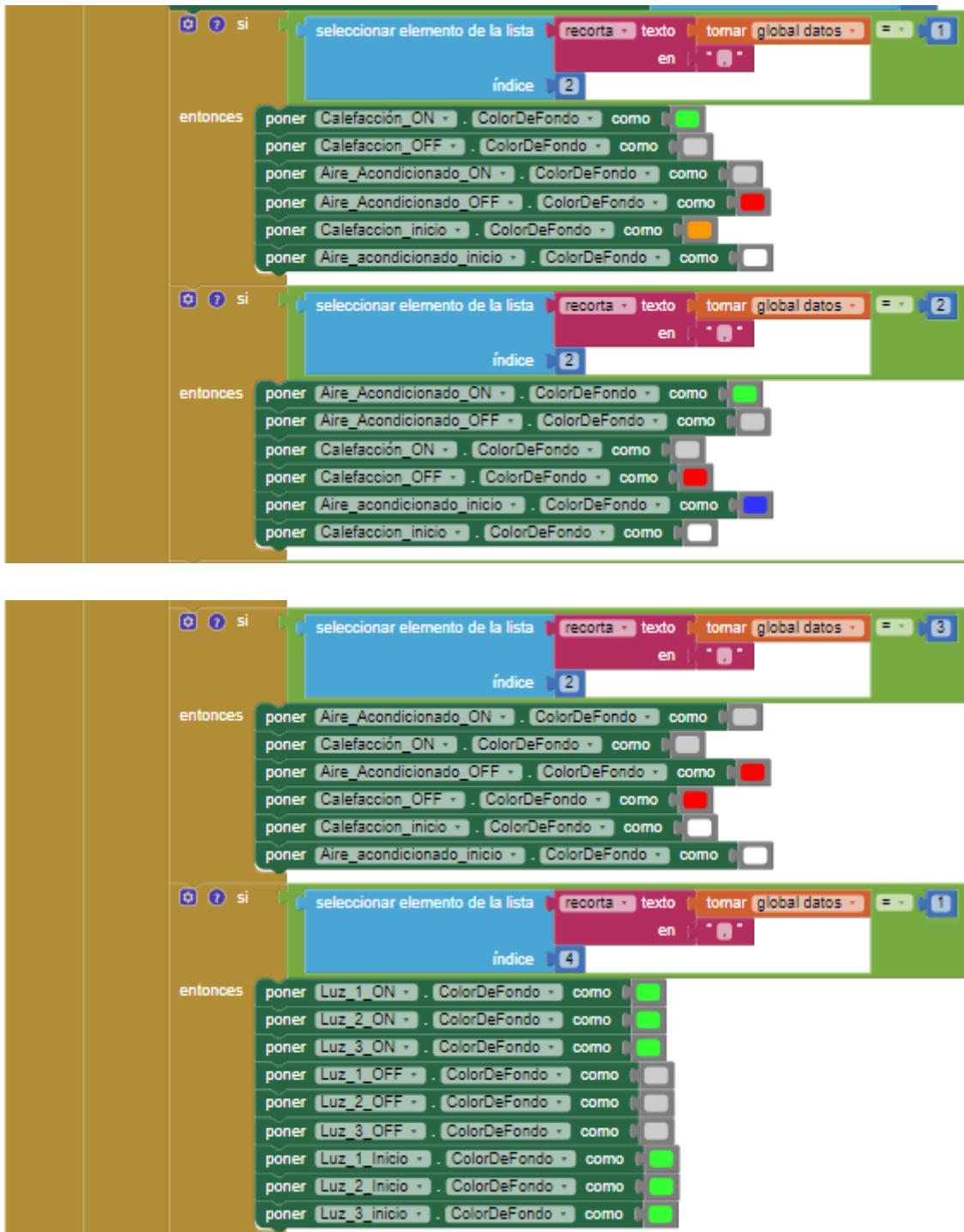
2. Código Mit App Inventor

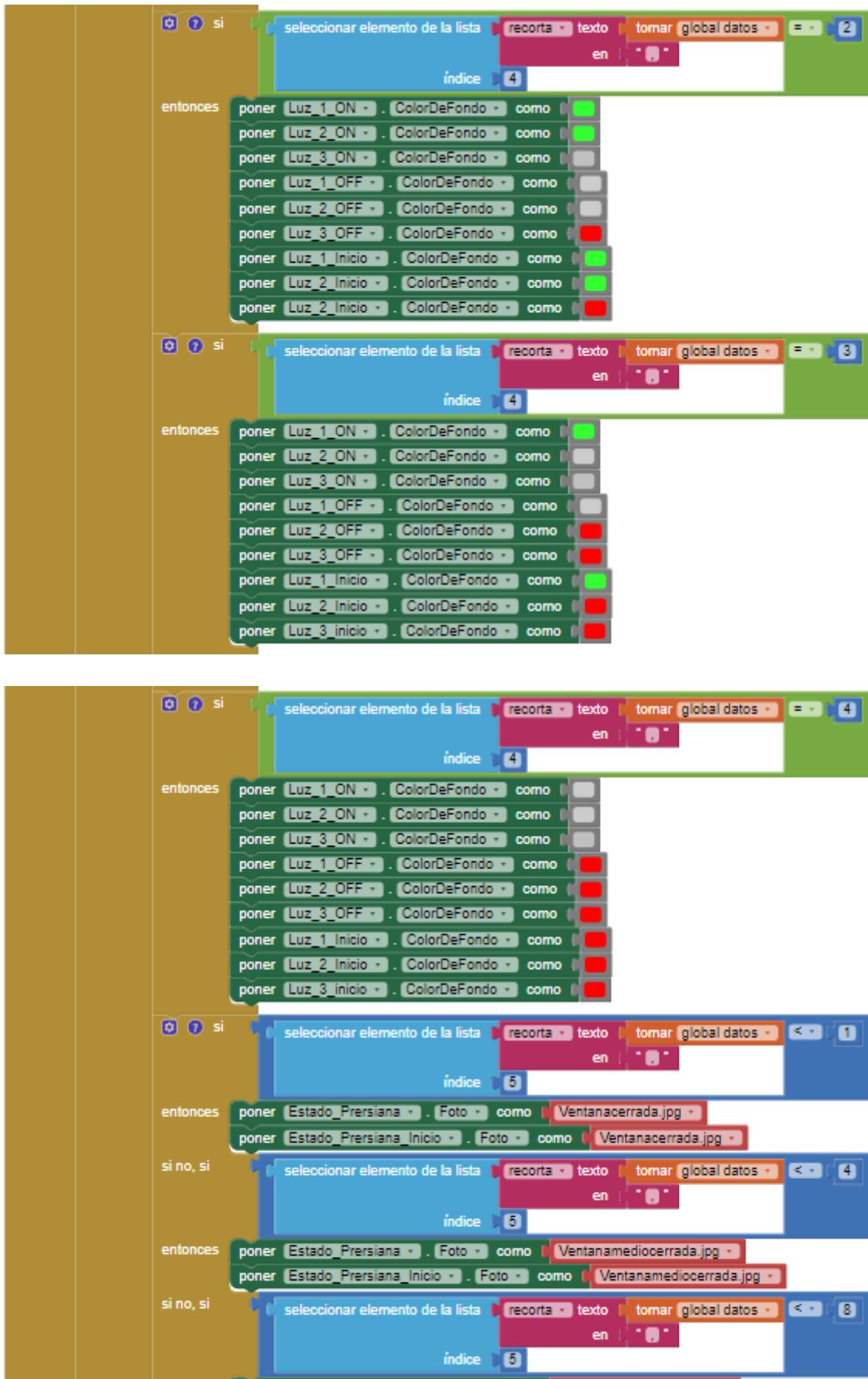
2.1 Selección bluetooth

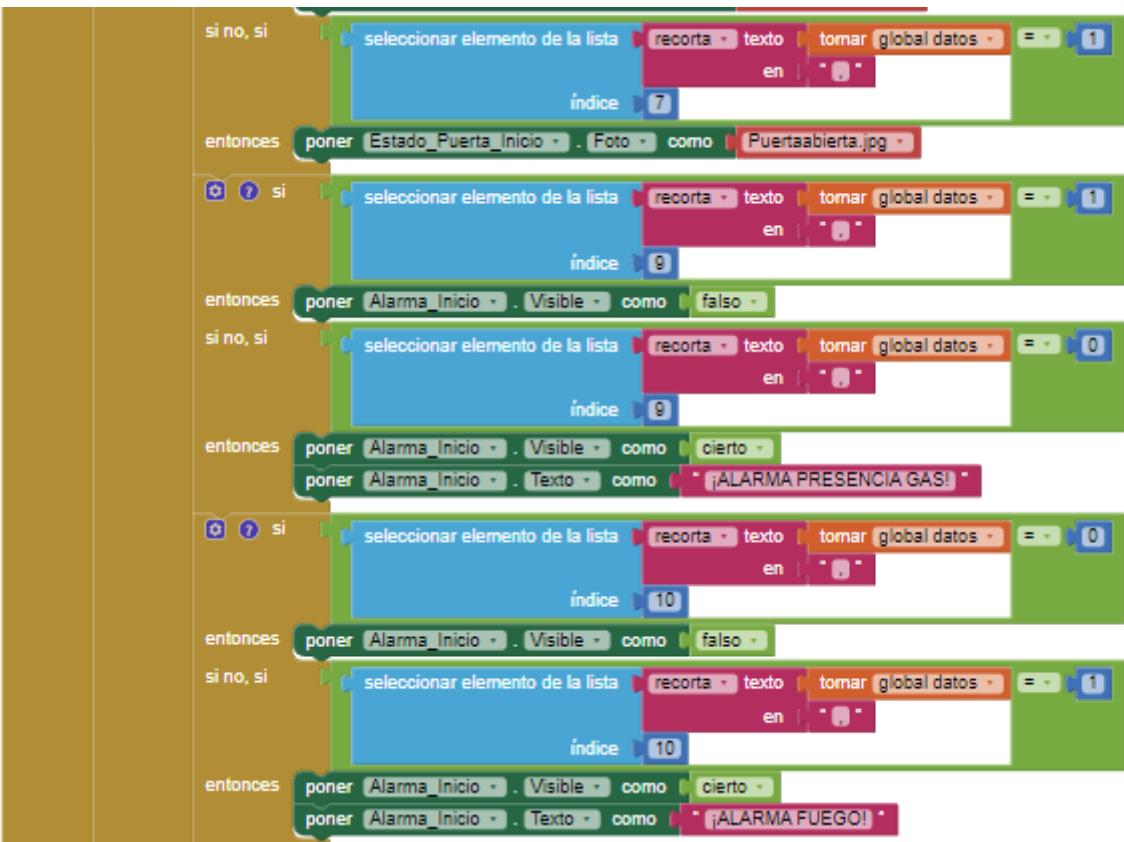
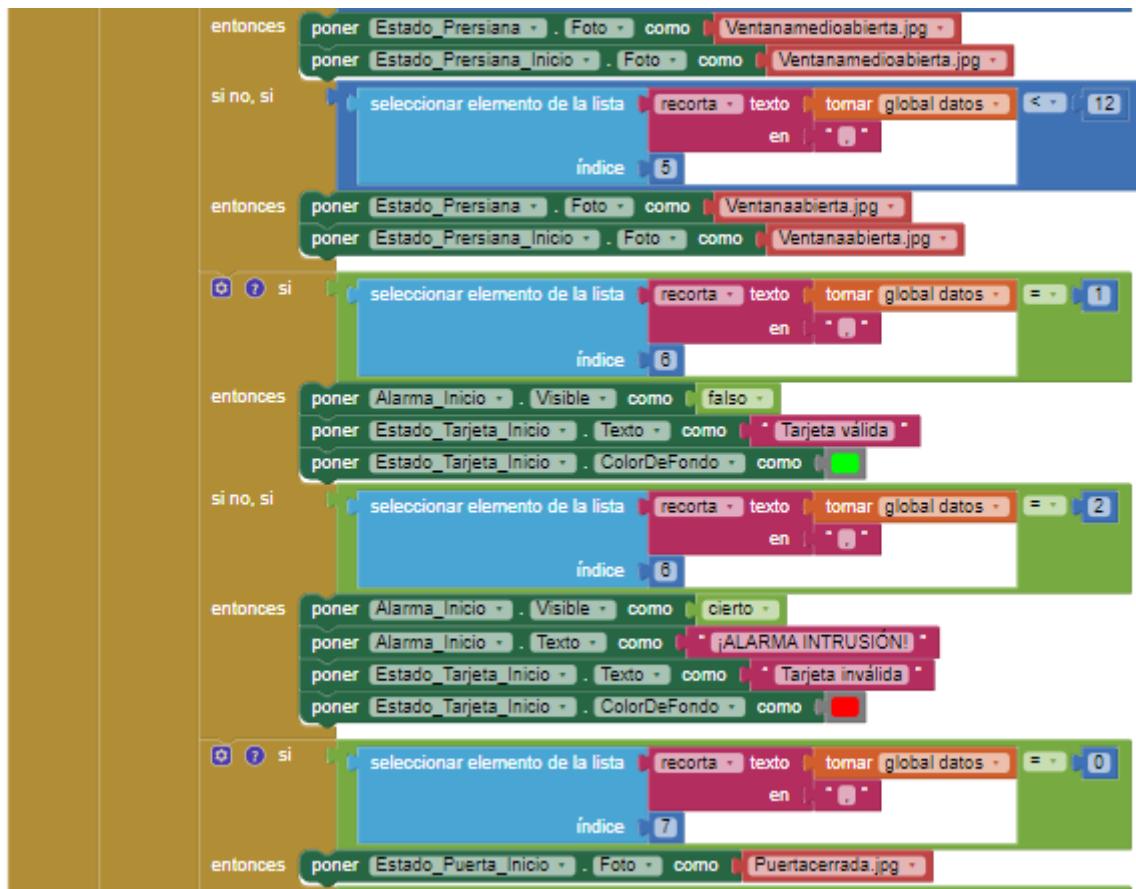


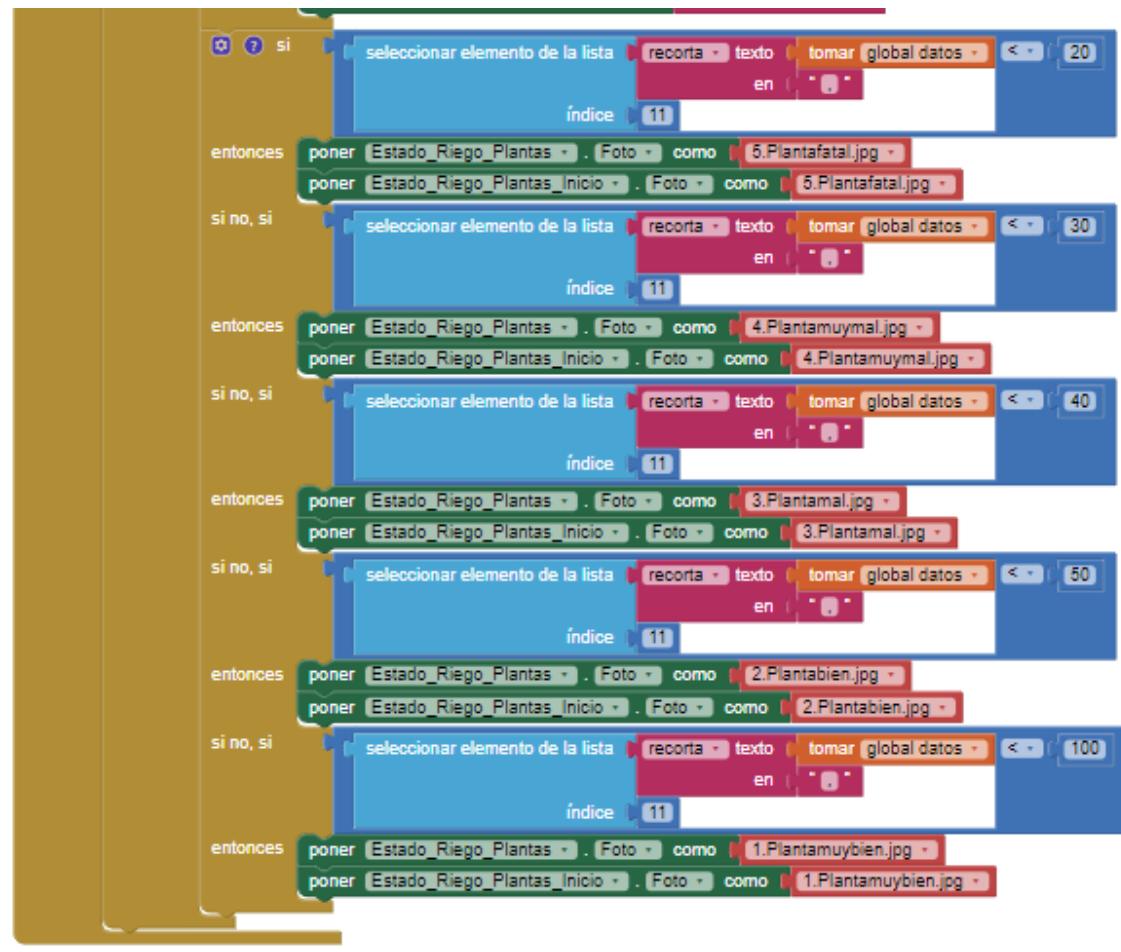
2.2 Recepción de datos a la aplicación por bluetooth



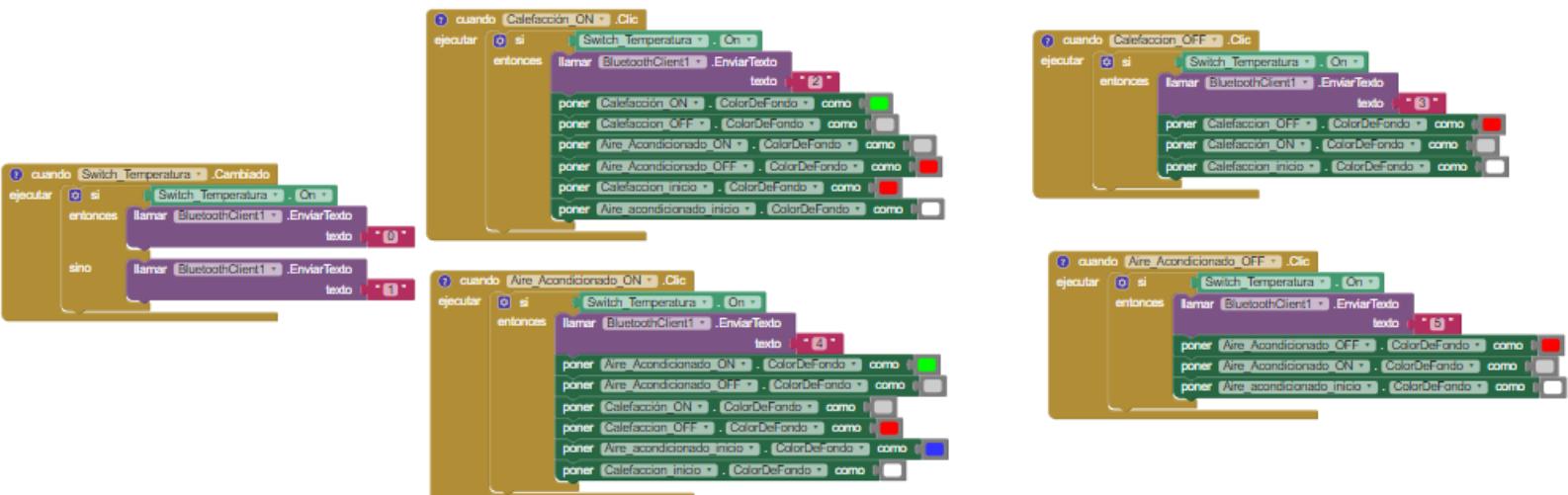




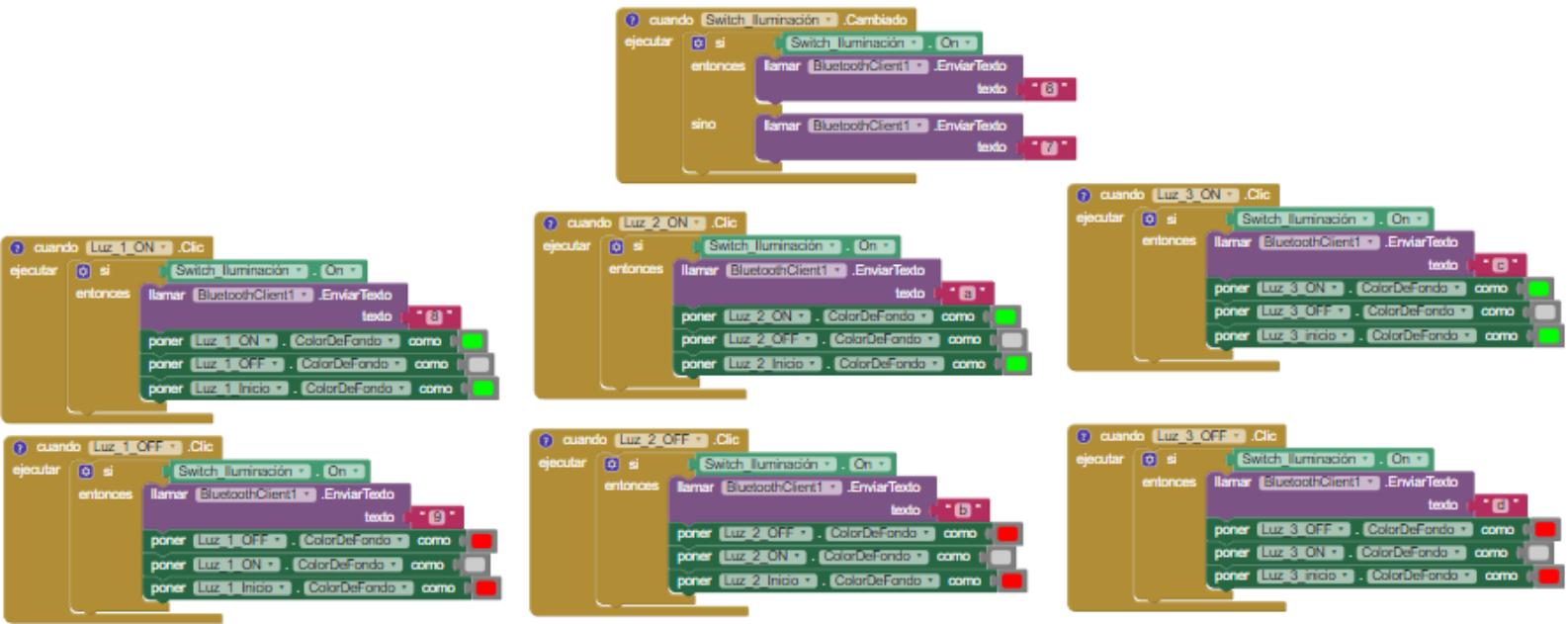




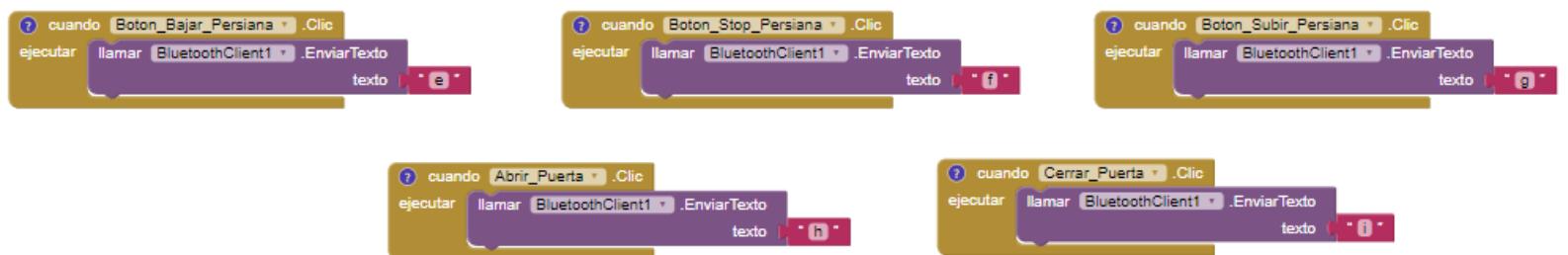
2.3 Envío de datos temperatura al microcontrolador



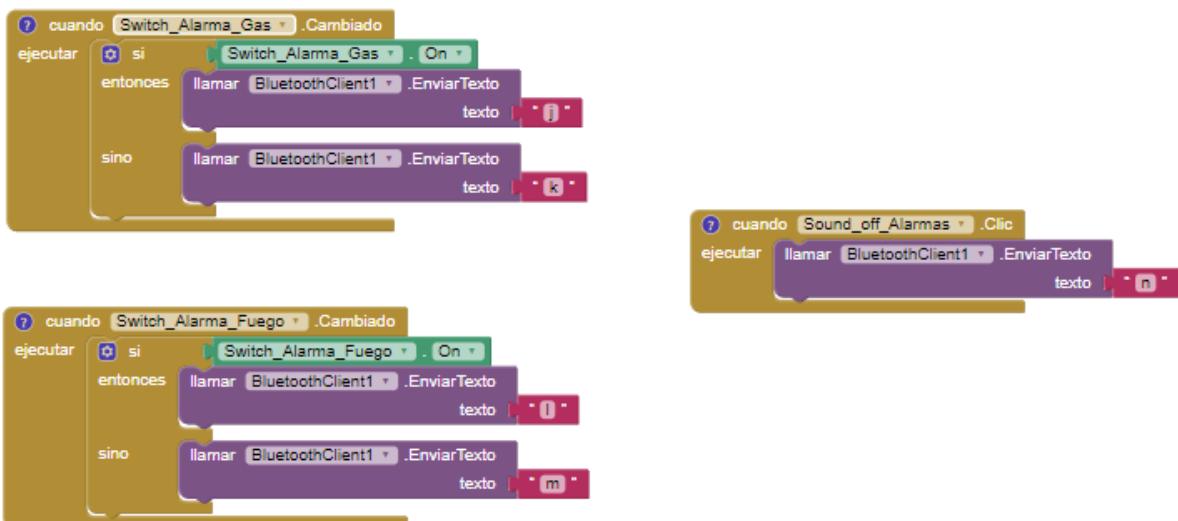
2.4 Envío de datos luces al microcontrolador



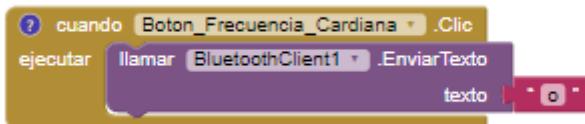
2.5 Envío de datos persianas y puertas al microcontrolador



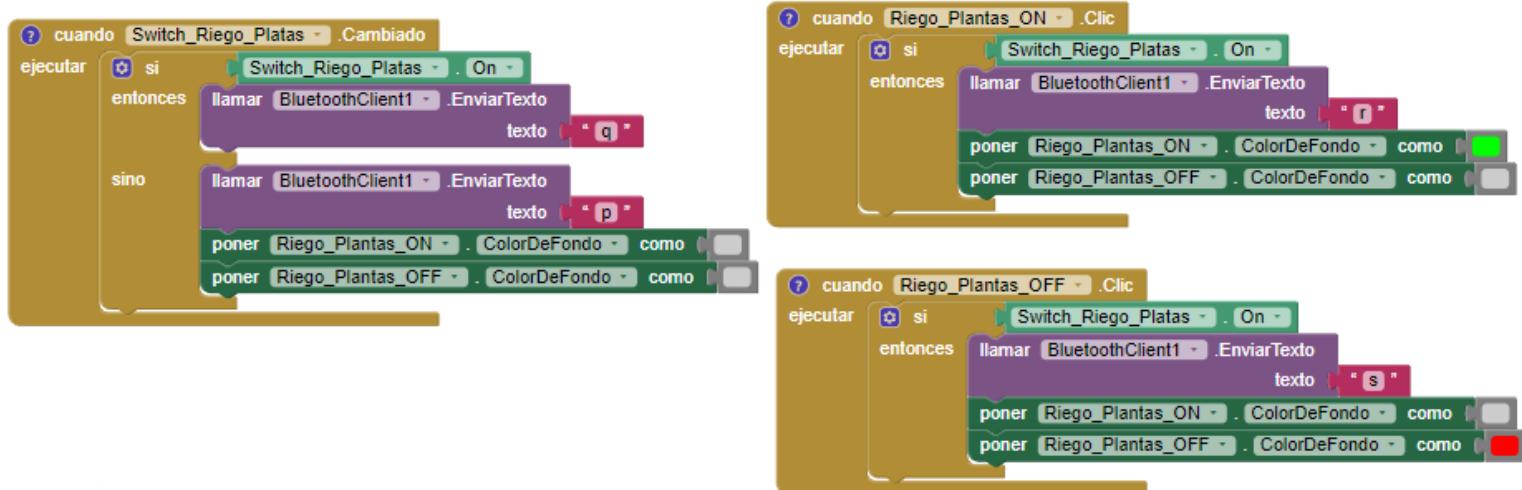
2.6 Envío de datos alarmas



2.7 Envío de datos frecuencia cardíaca



2.8 Envío de datos riego de plantas



2.9 Animación pantalla inicio

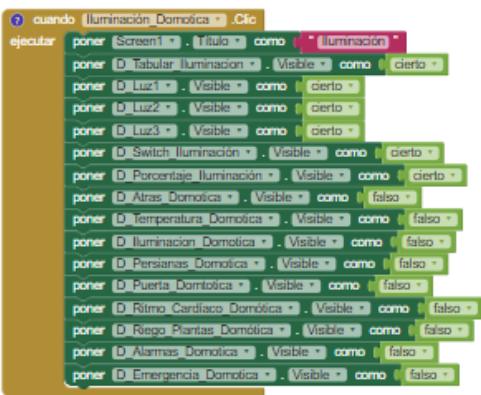




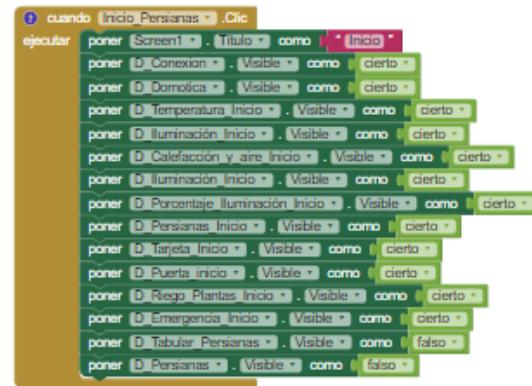
2.10 Animación pantalla temperatura



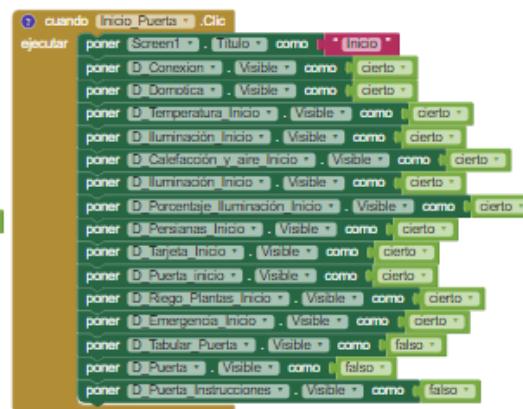
2.11 Animación pantalla iluminación



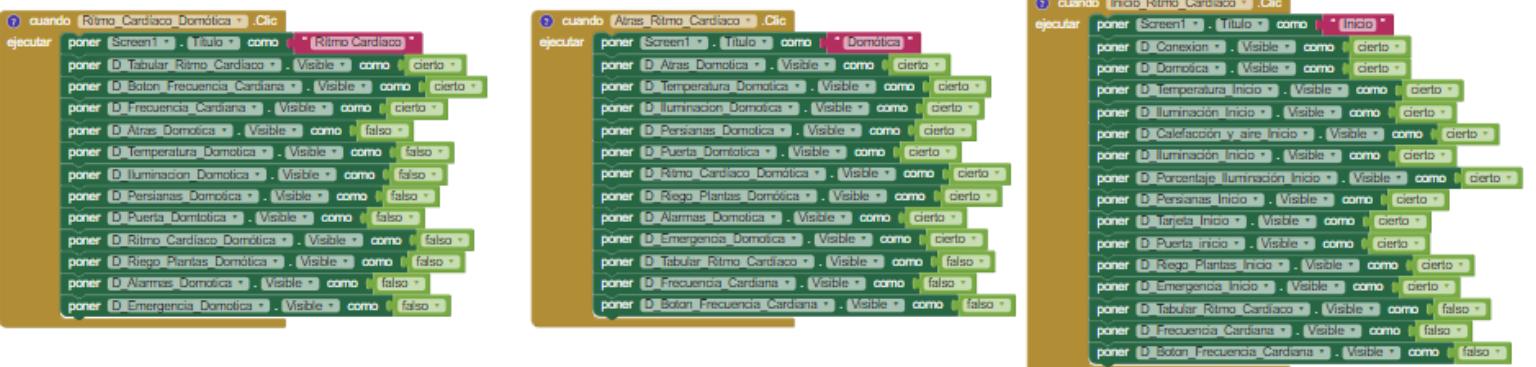
2.12 Animación pantalla persianas



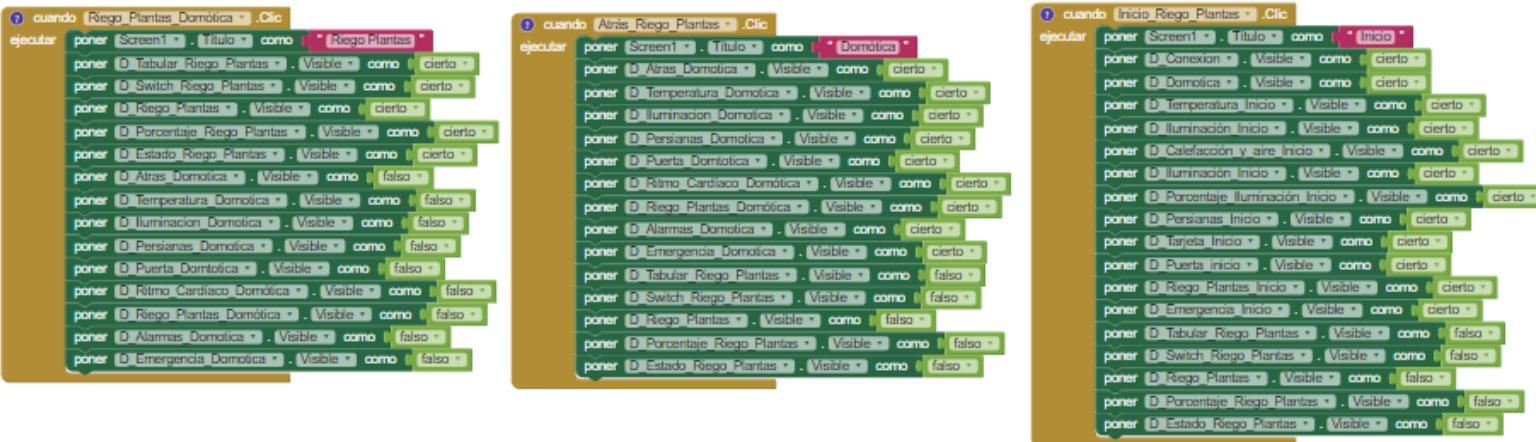
2.13 Animación pantalla puerta



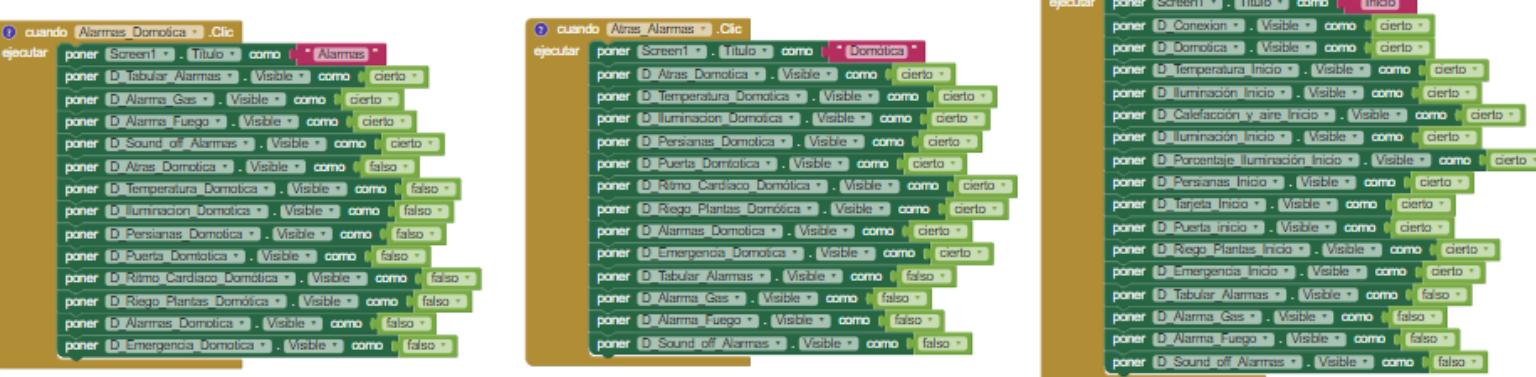
2.14 Animación pantalla ritmo cardíaco



2.15 Animación pantalla riego plantas



2.16 Animación pantalla alarmas



2.17 Animación pantalla emergencia

