

PowerDroid

Leonardo Bravo, Andrés Brito, Edwin Rodríguez

Abstract—En este documento se explicará como encender una luz por medio de un smarthphone con Android, por medio del proyecto "PowerDroid", detallando el diseño, la funcionalidad y además la tecnología empleada.

I. INTRODUCCIÓN

La realización de este proyecto se inspiró en la domótica, la cual es un conjunto de sistemas capaces de automatizar una vivienda, dando énfasis en la comodidad del usuario. El mismo, llamado "PowerDroid", consta de un dispositivo que se instala a la corriente en cualquier parte del hogar y mediante un webservice y una aplicación Android se pueden controlar los aparatos que tenga conectados. Otro uso sería encender y apagar luces de forma remota. Además, cuando la persona se encuentre en el hogar, puede operar el "PowerDroid" con un simple movimiento frente al dispositivo gracias a un sensor de movimiento.

II. DISPOSITIVOS Y TECNOLOGÍA EMPLEADA

Para realizar este proyecto, se necesitó utilizar:

- Placa de Arduino
- Laptop
- Internet
- Servo motor
- Sensor de movimiento
- Alargador de corriente
- Cable de red
- Cables
- Led
- Webservice
- Celular
- Caja
- Cinta

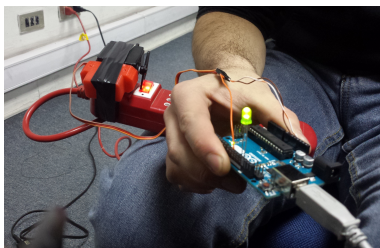


Fig. 1. Constucción del proyecto.

El trabajo además solicitó el uso de tres tópicos, los cuales se definirán a continuación.

Escuela de Informática y Telecomunicaciones

A. Servo GOTECK Continuous Rotation GS-3360BB

El servo es un motor que hace girar una hélice en 360°. Este dispositivo cuenta con tres hilos: alimentación, tierra, y señal. Para hacer funcionar el servo debe haber un cable de alimentación conectado al pin de 5V de la placa Arduino, otro cable a tierra se debe conectar al pin de tierra de la placa Arduino y finalmente un último cable debe ser conectado a un pin digital en la placa. Mediante un código se le envía la información como señal al motor y procederá a girar en la dirección y potencia que se le indique.^[1]



Fig. 2. Fotografía del Servo Motor.

B. Pir Sensor (Rev B)

Este pequeño sensor detecta el movimiento mediante la medición de los cambios en los niveles de infrarrojos (es decir, calor) emitida por los objetos cercanos. Cuando se detecta el movimiento del sensor PIR, emite una señal de alto en su pin de salida, señalando que ha detectado calor que para los efectos de este proyecto se traduce en movimiento. Normalmente detecta a una persona hasta aproximadamente 19 pies de distancia.

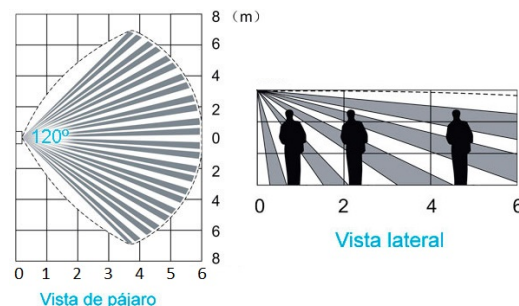


Fig. 3. Gráfica de sensibilidad del Pir Sensor

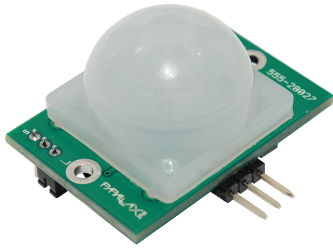


Fig. 4. Fotografía del Pir Sensor

C. Dispositivo Móvil

Este tópico consta de un aparato móvil, en este caso un smartphone con sistema operativo Android. Este dispositivo se conectará a una red inalámbrica con la cual tendrá comunicación con la placa Arduino mediante un webservice.

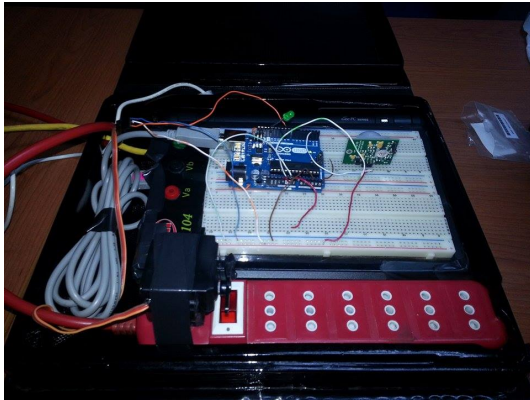


Fig. 5. Arquitectura del PowerDroid.

III. DISEÑO DEL SISTEMA

El sistema está compuesto por una placa de Arduino, la cual posee un sensor de movimiento y un servo motor. Cuando el sensor de movimiento se activa, el motor se moverá. Paralelamente a esto, habrá un dispositivo móvil el cual enviará órdenes al Arduino mediante un webservice para que mueva el motor de forma remota, así el usuario podrá controlar el "PowerDroid" desde cualquier lugar.

A continuación se definirán los distintos funcionamientos de las partes del sistema.

A. Funcionamiento Arduino

En el Arduino están ubicados el sensor de movimiento y el motor, y a su vez la placa está conectada a un computador que hace de servidor web. Cuando le llega la orden al Arduino desde el servidor, el motor se moverá, apagando el interruptor en caso de que se haya presionado OFF y encendiéndolo en caso de que sea un ON. Además, de forma independiente, el sensor de movimiento estará escuchando constantemente para activarse, entonces cuando haya movimiento, moverá el motor (encendiendo el interruptor si estaba apagado y viceversa. Además, si llega a encender o apagar el interruptor mediante

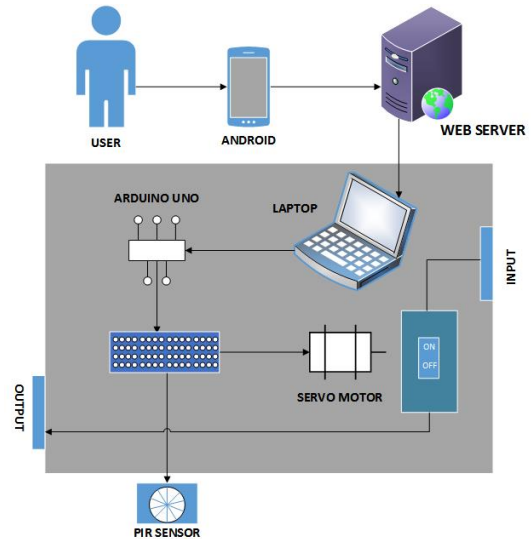


Fig. 6. Diagrama de la solución.

movimiento, habrá un delay de 3 segundos para no captar un exceso de movimiento, luego de ese delay se podrá darle una nueva instrucción al "PowerDroid" mediante movimiento.

B. Funcionamiento del webservice

El webservice es un programa desarrollado con Python Flask (que es un framework) que ejecuta instrucciones y programas (también hechos en Python). Estos programas ejecutados por el webservice, envían la orden de encendido o apagado de la luz según lo indicado por la aplicación móvil, por medio del puerto serial a la placa de Arduino. La comunicación entre la aplicación y el webservice es por medio de Json.

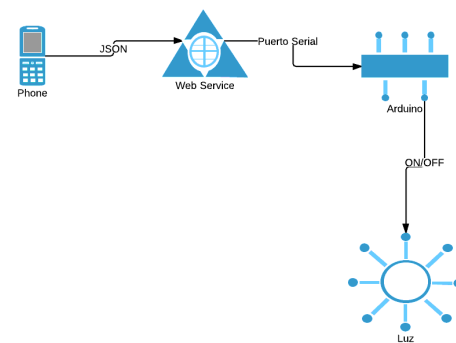


Fig. 7. Lógica de comunicación del sistema.

C. Funcionamiento de la aplicación móvil

La aplicación móvil fue desarrollada utilizando Phone-Gap, que es un framework que permite desarrollar aplicaciones móviles utilizando HTML5 y JQuery Mobile. Para este proyecto se compiló la aplicación utilizando Cordova: 7 para dispositivos Android. Consiste en tres vistas: la principal donde se encontrará el botón de encendido y apagado de las

luces, otra en la cual se podrá configurar la IP y el puerto del servidor y la última en la cual se encuentra la información relacionada al proyecto.

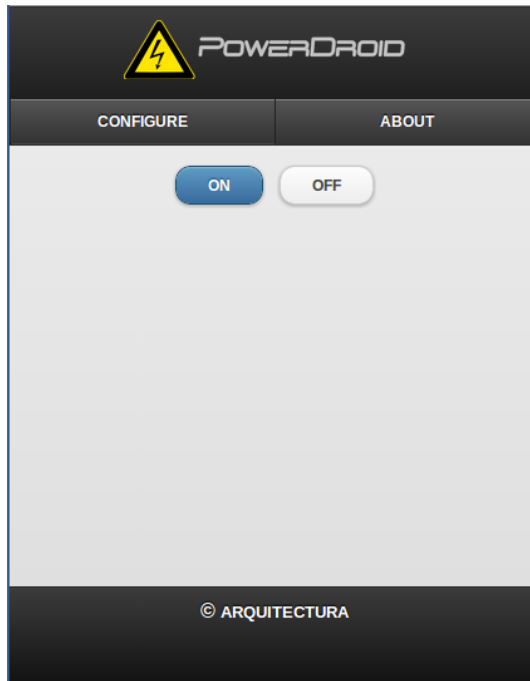


Fig. 8. Vista principal de la aplicación.

D. Código del Arduino

Debido a que se pidió comentado, se colocaron imágenes del código, pero de todas formas el código irá adjunto al correo.

```

1 // Definición de pines de salida y entrada
2 #define LED 13 // se define el led en la salida 13 del arduino
3 #include <Servo.h> // se incluye la libreria servo.h para el funcionamiento del servo motor.
4
5 Servo myservo; // declaramos myservo del tipo servo refiriendonos a nuestro servo motor.
6
7 int pos = 0; // declaramos pos tipo integer inicializado en 0
8 int estado = 0; // declaramos estado tipo integer inicializado en 0
9 int inputPin = 21; // Escogemos el pin del arduino de input 21 utilizado para el PIR sensor
10 int pirState = LOW; // Inicializamos el estado del PIR sensor, asumiendo que no hay movimiento detectado
11 int val = 0; // variable para lectura del pin de status
12
13 void setup() {
14   pinMode(inputPin, INPUT); // se declara el PIR sensor como input
15   Serial.begin(9600); //se inicializa el serial para la entrada de datos
16
17   myservo.attach(9); // la funcion es declarar que el servo motor estara conectado por el puerto 9 del arduino
18   pinMode(LED, OUTPUT); // se declara la variable led como output
19   myservo.write(90); //con esto le decimos al motor que no haga nada hasta espera respuesta del sensor o la aplicacion en android
20 }
21
22 void loop(){
23   //loop
24   //----- PIR SENSOR -----
25   val = digitalRead(inputPin); // lee el valor del input
26   if (val == HIGH) { // verifica si el input es HIGH
27     delay(150); // una espera de 150 ms
28
29     if (pirState == LOW) { // verifica si el estado del PIR esta en LOW
30       Serial.println("PIR ----> ENCENDIDO"); // imprime en el monitor del serial
31     }
32   }
33 }

```

Fig. 9.

```

42
43 pirState = HIGH; // se cambia el estado del PIR a HIGH
44 if (estado == 1) {estado = 0;} //ahora apaga
45 else if (estado == 0){estado = 1;} //ahora prende
46
47
48
49
50 //----- led -----
51 if (estado == 1) { // si el estado es igual a 1 se ejecuta el codigo
52   for (pos = 0; pos < 90; pos +=1) { //movimiento del motor
53     myservo.write(pos); // le envia el lo almacenado anteriormente al servo motor
54     delay(5); // espera 5 ms
55     myservo.write(90); // detiene el servo motor escribiendole 90
56   }
57   digitalWrite(LED, HIGH); // enciende la luz
58   Serial.println("LED && MOTOR----> ENCENDIDO"); // imprime en el monitor del serial
59   delay(3000); // espera de 3000 ms
60 } //mover motor a encendido
61
62 if (estado == 0) { // si el estado es igual a 0 se ejecuta el codigo
63   for (pos = 180; pos >= 90; pos -=1) { //movimiento del motor
64     myservo.write(pos); // le envia el lo almacenado anteriormente al servo motor
65     delay(5); // espera 5 ms
66     myservo.write(90); // detiene el servo motor escribiendole 90
67   }
68   digitalWrite(LED, LOW); //apaga la luz
69   Serial.println("LED && MOTOR----> APAGADO"); // imprime en el monitor del serial
70   delay(3000); // espera de 3000 ms
71 } //mover motor a apagado
72
73 } else {
74
75   delay(300); // espera de 3000 ms
76   if (pirState == HIGH) { // si el estado es igual a high ejecuta el codigo
77     // we have just turned of
78     Serial.println("PIR ----> APAGADO"); // imprime en el monitor del serial
79   }
80 }

```

Fig. 10.

```

81 // Definición de pines de salida y entrada
82 #define LED 13 // se define el led en la salida 13 del arduino
83 #include <Servo.h> // se incluye la libreria servo.h para el funcionamiento del servo motor.
84
85 Servo myservo; // declaramos myservo del tipo servo refiriendonos a nuestro servo motor.
86
87 int pos = 0; // declaramos pos tipo integer inicializado en 0
88 int estado = 0; // declaramos estado tipo integer inicializado en 0
89 int inputPin = 21; // Escogemos el pin del arduino de input 21 utilizado para el PIR sensor
90 int pirState = LOW; // Inicializamos el estado del PIR sensor, asumiendo que no hay movimiento detectado
91 int val = 0; // variable para lectura del pin de status
92
93 void setup() {
94   pinMode(inputPin, INPUT); // se declara el PIR sensor como input
95   Serial.begin(9600); //se inicializa el serial para la entrada de datos
96
97   myservo.attach(9); // la funcion es declarar que el servo motor estara conectado por el puerto 9 del arduino
98   pinMode(LED, OUTPUT); // se declara la variable led como output
99   myservo.write(90); //con esto le decimos al motor que no haga nada hasta espera respuesta del sensor o la aplicacion en android
100 }
101
102 void loop(){
103   //loop
104   //----- PIR SENSOR -----
105   val = digitalRead(inputPin); // lee el valor del input
106   if (val == HIGH) { // verifica si el input es HIGH
107     delay(150); // una espera de 150 ms
108
109     if (pirState == LOW) { // verifica si el estado del PIR esta en LOW
110       Serial.println("PIR ----> ENCENDIDO"); // imprime en el monitor del serial
111     }
112   }
113 }

```

Fig. 11.

IV. CONCLUSIÓN

En resumen el dispositivo “PowerDroid” es muy útil, pues ayuda directamente al control de dispositivos eléctricos desde fuera del hogar, y también a un control fácil y rápido cuando se está dentro del hogar. “PowerDroid” podría mejorar de muchas formas, por ejemplo:

- 1) Reemplazar el motor que apaga y enciende el interruptor por un relay.
- 2) Controlar más de un circuito eléctrico, ya que actualmente administra solo una conexión de energía.
- 3) Reemplazar el puerto serial con el cual se conecta el Arduino por una placa wireless, conectando el arduino a la red de forma inalámbrica.
- 4) Agregar las redes eléctricas correspondientes a los distintos sectores del hogar a la aplicación del celular, permitiendo administrar los lugares deseados.
- 5) Agregar otros sensores como por ejemplo de temperatura, para un mayor control de la casa.
- 6) Colocar una pantalla al “PowerDroid” para mostrar el estado actual del dispositivo.

V. REFERENCIAS

- 1) Arduino página oficial
[<http://arduino.cc/es/Reference/Servo>]
- 2) Parallax inc.
[<http://www.parallax.com/product/555-28027>]
- 3) Código del webservice.
[<https://github.com/l30bravo/toys/tree/master/PYTHON>
]
- 4) Aplicación PowerDroid Android.
[https://github.com/l30bravo/toys/tree/master/APPS/Arduino_uno]