CONTROL DE ILUMINACION DE FORMA INALÁMBRICA CON ARDUINO Y ANDROID

SEBASTIAN RIVERA GALVIS
CÓDIGO: 1088305963

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE TECNOLOGÍA
PROGRAMA DE TECNOLOGÍA ELÉCTRICA
PEREIRA
2015

CONTROL DE ILUMINACION DE FORMA INALÁMBRICA CON ARDUINO Y ANDROID

SEBASTIAN RIVERA GALVIS
CÓDIGO: 1088305963

Trabajo de grado presentado como requisito para optar por el título de Tecnólogo en Electricidad

DIRECTOR
Carlos Alberto Ríos Porras
Profesor asistente
Universidad Tecnológica de Pereira

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE TECNOLOGÍA
PROGRAMA DE TECNOLOGÍA ELÉCTRICA
PEREIRA
2015

CONTENIDO

pág.

LISTA DE CUADROS

pág.

LISTA DE FIGURAS

GLOSARIO

App Inventor: es un entorno de desarrollo integrado (IDE) para la plataforma Android. Fue anunciado por Ellie Powers el 16 de mayo de 2013. Android Studio está disponible para desarrolladores para probarlo gratuitamente (1).

Arduino: es una plataforma de hardware libre, basada en una placa con un micro controlador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares (2).

Aplicación: se incluyen correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Estas aplicaciones están escritas en lenguaje de programación Java.

Bluetooth: es una especificación industrial para redes inalámbricas de área personal (*Significado en inglés* - WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz (3).

Domótica: se llama domótica al conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación

Relé: dispositivo electromagnético que, estimulado por una corriente eléctrica muy débil, abre o cierra un circuito en el cual se disipa una potencia mayor que en el circuito estimulador.

Plataforma Android: Es una plataforma de software para dispositivos móviles que incluye un Sistema Operativo y aplicaciones de base (4)

Led: Sigla de la expresión inglesa light-emitting diode, ‘diodo emisor de luz’.

USB: periférico que permite conectar diferentes periféricos a una computadora.

RESUMEN

En este proyecto se presenta un control de iluminación a distancia por medio de un dispositivo con plataforma Android, desarrollado bajo la necesidad de mostrar a los estudiantes del programa de Tecnología Eléctrica que se puede combinar el conocimiento de la carrera con las nuevas tecnologías como lo es la plataforma Android y así crear nuevos proyectos. Además, se basa en la introducción de un nuevo dispositivo para beneficiar a las usuarios de los dispositivos Android, dándoles una aplicación y un artefacto dinámico, con el cual puedan controlar los artefactos de su hogar, esto implica control, confort y seguridad hacia las personas. Un detalle importante que surgió en la realización de este proyecto fue la manera de integrar la tecnología, para que personas con capacidades limitadas puedan realizar actividades comunes en una vivienda, como el encendido de luminarias, control de acceso y seguridad, es decir que el proyecto sea beneficioso y ayudar a personas con discapacidad auditiva, visual o motora.

Palabras clave: Android, seguridad.

INTRODUCCION

La tecnología se desarrolla de manera impresionante y cada vez surgen nuevos inventos que intentan facilitar la vida del hombre. Sin embargo, el hombre se ha encontrado con pequeñas dificultades que un dispositivo celular podría resolver a distancia, online o automáticamente. Algunas de estas tareas son: controlar las luminarias de una residencia o centro estudiantil, controlar los artefactos de una cocina, controlar las puertas de acceso de una residencia o controlar el encendido del aire acondicionado. En estas actividades es importante integrar la comodidad del control a distancia y que mejor manera de hacerlo que un dispositivo como el teléfono celular, acompañado de una de las plataforma de vanguardia como lo es Android.

Se llama demótica al conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad, desde dentro y fuera del hogar.

Son muchos los factores que pueden caracterizar la forma de vida de las personas, por ejemplo, población con algún tipo de discapacidad y que viven de forma independiente o adultos mayores que viven solos; labores tan comunes como abrir o cerrar una ventana o una puerta se convierten en un problema para estas personas; pero con la implementación de la demótica se disminuye el grado de dificultad para realizar estas tareas.

En este proyecto se combina la nueva plataforma de programación Android con el conocimiento de la electrónica que se enseña en el programa de Tecnología Eléctrica para crear un dispositivo que controle luminarias a distancia desde la comodidad de un teléfono inteligente.

Con relación a la temática de microprocesadores Arduino y la plataforma Android, en Pereira se han desarrollado los siguientes trabajos de grado:

- En (5) se implementó un proyecto para conocer la estructura y las diferentes herramientas hardware y software utilizadas para el desarrollo de dispositivos demóticos para el hogar, por medio de la plataforma de hardware libre, microcontrolador Arduino UNO, con el fin de generar un ambiente más amigable para los desarrolladores que deseen introducirse en ésta nueva tendencia y de dar a conocer los grandes beneficios que puede brindar a la sociedad. Proyecto realizado en la Universidad Católica de Pereira.

- En (6) se implementó un control de un robot a distancia por medio de Bluetooth y una aplicación de Android con el objetivo de explorar con este robot lugares o terrenos de difícil acceso para el hombre, o simplemente explorar a distancia sin necesidad de la presencia de una persona. Proyecto realizado en la Universidad Nacional Autónoma de México.

- En (7) se realizó una guía del sistema operativo Android explicando todas sus características y funcionalidades para dispositivos móviles. Proyecto realizado en la Universidad Tecnológica de Pereira.

- En (8) se implementó un trabajo de grado se desarrollan los conceptos básicos relacionados con la plataforma ARDUINO Uno R3. Esto como complemento académico del curso Sistemas Digitales II. Este trabajo contiene como información base una conceptualización del ARDUINO Uno R3, el funcionamiento de la placa y sus características técnicas. Además, se presenta en detalle algunos de los protocolos más utilizados para la programación y comunicación serial del ARDUINO. Proyecto realizado en la Universidad Tecnológica de Pereira.

- En (9) se realizó un trabajo que tuvo como objetivo el diseño y desarrollo de un prototipo basado en una placa de desarrollo Arduino AT Mega 2560, que permita por medio de un joystick controlar el desplazamiento y giro de una silla de ruedas ultraliviana de bajo costo fabricada con tubería de PVC. Proyecto realizado en la Universidad Tecnológica de Pereira.

- En (10) se implementó un proyecto que surge al ver la necesidad de una aplicación para que las personas con limitación visual puedan jugar ajedrez en plataformas móviles con sistema operativo Android. Este trabajo contiene como información base una conceptualización del ARDUINO Uno R3 el funcionamiento de la placa y sus características técnicas. Además, se presenta en detalle algunos de los protocolos más utilizados para la programación y comunicación serial del Arduino. Proyecto realizado en la Universidad Tecnológica de Pereira.

OBJETIVOS

El objetivo general de este proyecto consiste en controlar la iluminación de forma inalámbrica con Arduino Uno y Android.

Los Objetivos Específicos son:

- Aprender el funcionamiento del software App Inventor y de la plataforma Arduino
- Diseñar la aplicación para el control de luminarias con el software App Inventor
- Diseñar los circuitos eléctricos y electrónicos para el control de las luminarias y de la plataforma Arduino
- Realizar pruebas con los circuitos y programas diseñados para el control inalámbrico de las luminarias por medie de conexión Bluetooth.

Alcances y limitaciones.

Los alcances de este proyecto en la aplicación Android, en el programa Arduino son: Está diseñada para controlar el encendido y el apagado de una luminaria, o controlar un ramal completo de iluminarias tipo Led, Esta aplicación con respecto al módulo Arduino uno R3 y la conexión Bluetooth tiene un alcance de 20 - 25 metros suficientes para cubrir una casa familiar y tener un control de su iluminación.

1. SOFTWARE APP INVENTOR

¿Qué es App Inventor?

App Inventor es una plataforma de Google Labs para crear aplicaciones de software para el sistema operativo Android. De forma visual y a partir de un conjunto de herramientas básicas, el usuario puede ir enlazando una serie de bloques para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web. Las aplicaciones de App Inventor están limitadas por su simplicidad, aunque permiten cubrir un gran número de necesidades básicas en un dispositivo móvil (11).

En App Inventor se diseña la aplicación o app a través de un entorno de desarrollo gráfico. Después para programar se usa un lenguaje de programación basado en un lenguaje visual a partir de bloques Java. Las librerías de estos bloques han sido desarrolladas por el MIT y son de uso libre. En estos bloques se encuentran funciones, sentencias y elementos muy comunes en la mayoría de lenguajes de programación. Gracias a esto se pueden crear aplicaciones apps de Android de una manera rápida y sencilla sin que haya la necesidad de tener muchos conocimientos de Java. Una vez finalizado el diseño y la programación de la app, se descarga y se instala en un dispositivo Android o también se puede probar en el emulador que proporciona App Inventor (11).

Para suscribirse al servicio de App Inventor se necesita una cuenta de google. El App Inventor puede ser accedido desde la siguiente dirección web:

http://appinventor.mit.edu/explore/

El App Inventor está conformado por tres herramientas:

- Gestor de proyectos
- Diseñador
- Editor de bloques

1.1. GESTOR DE PROYECTOS

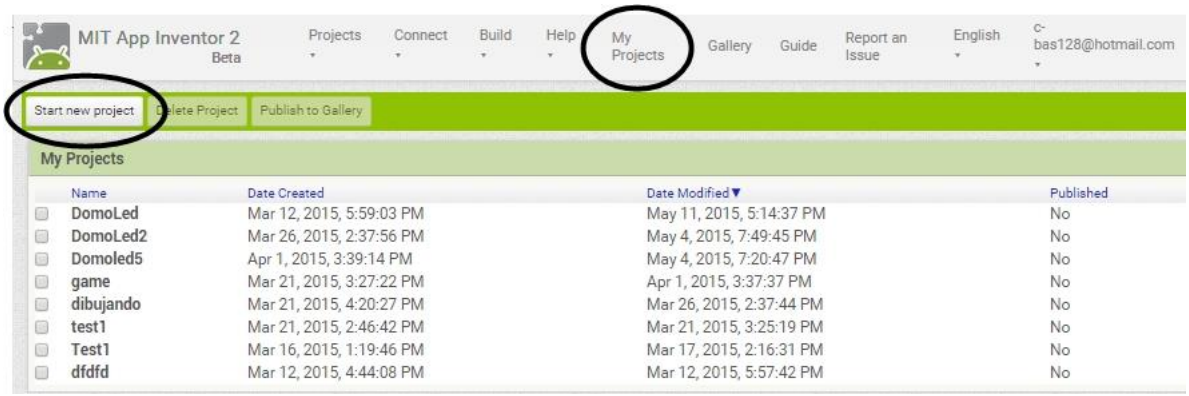
El gestor de proyectos es donde se guardan nuestros proyectos.

Lo primero que se debe tener, es una cuenta Google para después ingresar a la página http://appinventor.mit.edu/explore/ al lado derecho se encuentra un botón que dice “Create” allí se le otorga el permiso con su cuenta Google para proseguir al programa. Ver la Figura 1.

Figura 1. Botón para acceder al programa.

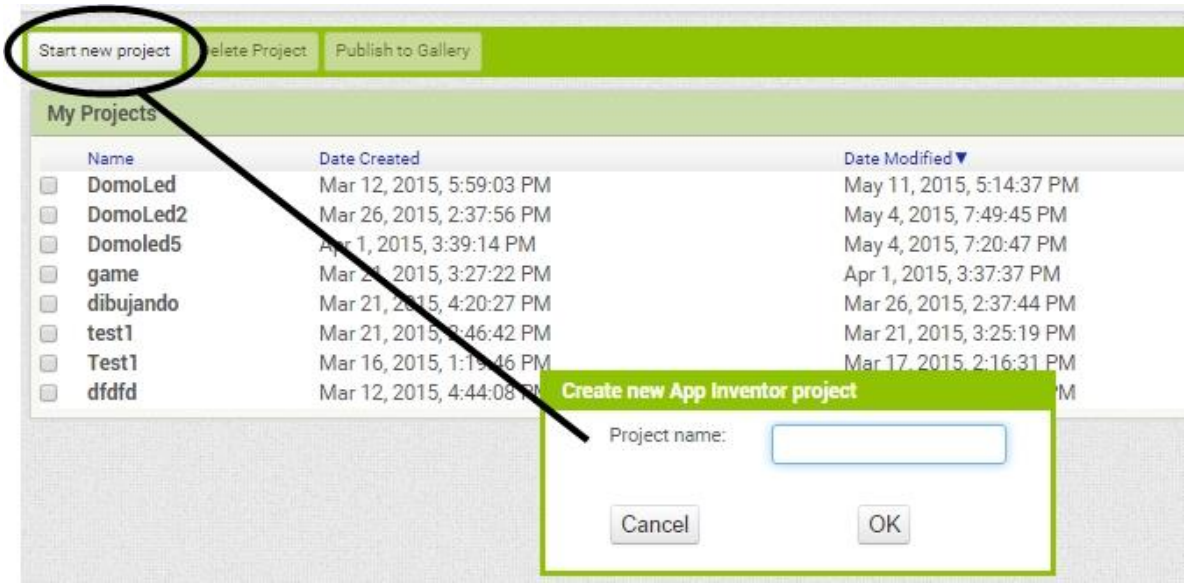
En el gestor de proyectos se selecciona “My Projects”, ahí se pueden observar diferentes opciones de cómo crear un nuevo proyecto (Star new project), eliminar proyecto (Delete Project) y publicar en la galería (Publish to Gallery), ver la Figura 2.

Figura 2. Gestor de proyectos



Se selecciona “Star new project” para crear un nuevo proyecto. Saldrá una ventana para ingresar el nombre del proyecto que el usuario le quiera dar, ver la Figura 3.

Figura 3. Asignar nombre del proyecto.



1.2. DISEÑADOR

Se trata de la ventana del diseñador en la que se construye, mediante el ratón la parte estética de la aplicación Android ver la Figura 4.

- La paleta (palette): contiene todos los elementos que se pueden insertar en la aplicación. Hay elementos gráficos como cuadros de texto, botones, lienzo de dibujo (Canvas) y elementos que no se ven en la pantalla del móvil, como base de datos (TinyDB), acelerómetro, cámara de vídeo, etc, ver la Figura 4.

- Visor (viewer): El visor de la pantalla, simula la apariencia que tendrá la aplicación en el móvil. Para añadir un elemento a la pantalla hay que arrastrarlo desde la paleta y soltarlo en el visor. Los elementos que no tengan visibilidad se arrastran también al viewer y automáticamente se desplazarán debajo de él bajo el epígrafe “Non-visible components”, ver la Figura 4.

- Componentes (Components): muestra la lista de los componentes que se han colocado en el proyecto. Cualquier componente que haya sido arrastrado y soltado desde la paleta al visor aparecerá ahí. Si se quiere borrar alguno es en la lista de componentes donde está el botón que permite borrarlo, ver la Figura 4.

- Medio de comunicación (media): muestra las distintas imágenes y sonidos que estarán disponibles para el proyecto. Cualquier archivo de imagen o audio que se quiera usar en la aplicación hay que insertarlo usando este apartado para que esté disponible, ver la Figura 4.

- Properties (propiedades): cada vez que en el Viewer se seleccione un componente, en Properties aparecerán todos los detalles que se puedan cambiar de ese componente. Por ejemplo, al hacer clic sobre un componente TextBox se podrá cambiar en Properties su color, texto, fuente, etc, ver la Figura 4.

Figura 4. Herramienta de diseño.

En la Figura 4 se observan 6 partes bien diferenciadas:

La barra superior. Las opciones de esta barra son:

- **Add Screen:** Añade una nueva ventana actual de la aplicación.
- **Remove Screen**: Elimina la ventana actual de la aplicación.
- **Blocks:** Al pulsar en este botón, descargará el lanzador de Blocks editor y este arrancará.
- **Designer:** Área de diseño estético de la aplicación.

A la izquierda, el menú **Palette** (paleta):

- **User interface:** da acceso a los componentes más básicos de la interfaz como botones, imágenes, etiquetas etc.
- **Layout:** permite organizar los elementos por la interfaz, debido a la limitación de App inventor se tendrá que utilizar estos componentes para que todos los elementos no estén únicamente en vertical.
- **Media:** Permite animar imágenes y que puedan desplazarse por la pantalla.
- **Drawing and Animation:** permite animar imágenes y que puedan desplazarse por la pantalla.
- **Sensors:** permite usar diferentes funciones del acelerómetro de un dispositivo.
- **Connectivity:** permite realizar conexiones bluetooth o conexión con internet.

En el centro, **Viewer:**

- En la parte superior de esta ventana, se observan las diferentes pantallas de la aplicación.
- En el centro se puede ver la vista previa de la aplicación. Arrastra componentes de las ventanas de **Palette** al **Viewer** para incluirlas en la aplicación.

En la Figura 4 se encuentra la pestaña **Components**. Todos los componentes que se incluyan aparecerán aquí y permitirá cambiarles el nombre, borrarlos o editarlos.

En la Figura 4 se encuentra **Properties**, para editar las opciones de los componentes, como modificar su forma.

En la Figura 4 está la opción **Media**, a partir de la cual se pueden subir diferentes componentes multimedia, como imágenes, sonidos, etc., y así acceder a ellos rápidamente.

1.3. EDITOR DE BLOQUES

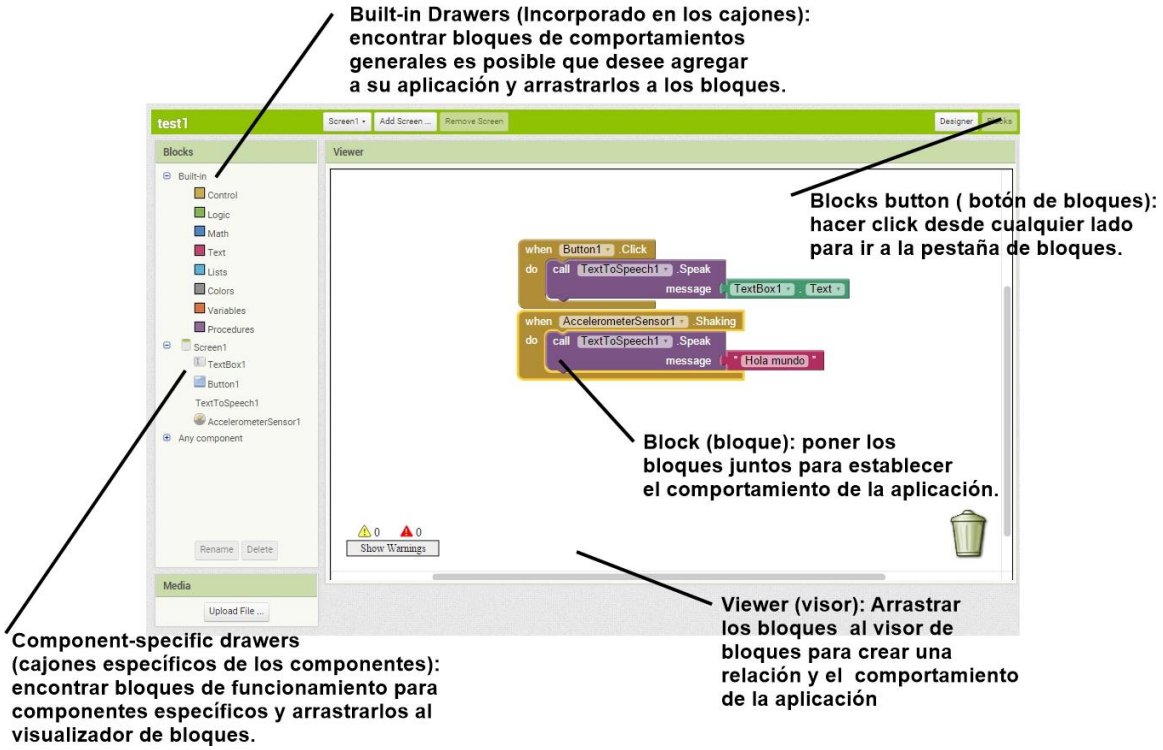
Donde se programan las acciones que se realizan con la interfaz anteriormente diseñada, ver la sección 1.2. DISEÑADOR.

El comportamiento de la App se programa mediante bloques o piezas **(Built-in bloks)** en el editor de bloques, ver la Figura 5.

Built-in blocks: los bloques incorporados están disponibles independientemente de qué componentes están en el proyecto. Además de estos bloques de lenguaje, cada

componente del proyecto tiene su propio conjunto de bloques específicos de sus propios actos, métodos y propiedades. Ver la Figura 5. Editor de bloques.Figura 5.

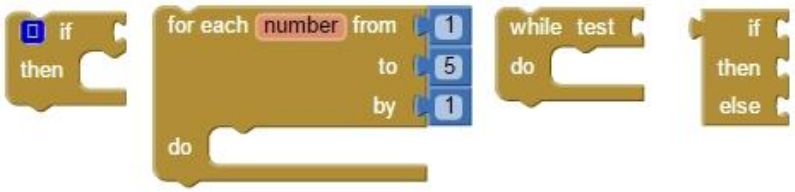
Figura 5. Editor de bloques.



1.3.1. Bloques de control

Con frecuencia se necesita realizar distintas acciones en función de que ocurra o no algo. En App Inventor para condicionar el programa se tienen algunas estructuras if-then, if-then-else, while y for each, como se muestra en la Figura 6.

Figura 6. Bloques de control.



If-then: Prueba una condición dada. Si la condición es verdadera, realiza las acciones en una determinada secuencia de bloques; de lo contrario, se ignoran los bloques.

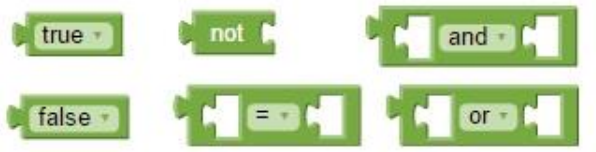
For each from to: Ejecuta los bloques en la sección de tareas para cada valor numérico a partir de **from** y termina en **to,** incrementando el número por el valor de **by** cada vez que se utiliza el nombre de la variable dada.

While: Pruebas de la condición **test**. De ser cierto, lleva a cabo la acción indicada en **do**, luego prueba de nuevo. Cuando la prueba es falsa, los extremos de bloque y la acción que figuran en **do** ya no se realizan.

If then else: Prueba una condición dada. Si el resultado es cierto, lleva a cabo las acciones en la secuencia do de bloques; de lo contrario, lleva a cabo las acciones en la secuencia **else** de bloques.

1.3.2. Bloques lógicos

Figura 7. Bloques lógicos.



True: Representa el valor constante de la verdad. Se usa para ajustar los valores de propiedad booleanas de los componentes, o como el valor de una variable que representa una condición.

False: Representa el falso valor constante. Se usa para ajustar los valores de propiedad booleanas de los componentes, o como el valor de una variable que representa una condición.

Not: Realiza negación lógica, volviendo falsa si la entrada es verdadero, y verdadero si la entrada es falsa.

Equal and not equal: Comprueba si los argumentos son iguales o no lo son.

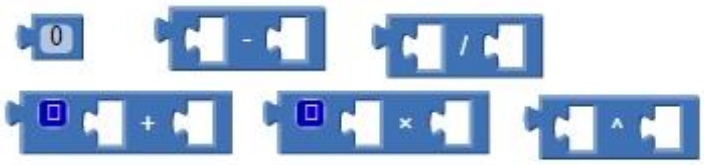
And: Comprueba si todas las condiciones lógicas son verdaderas. El resultado es verdadero si y sólo si se cumplen todas las condiciones.

Or: Comprueba si alguno de un conjunto de condiciones lógicas son verdaderas. El resultado es verdadero si una o más de las condiciones ensayadas son ciertas.

1.3.3. Bloques de matemáticas:

En algunas ocasiones se necesita usar aplicaciones matemáticas en algunas aplicaciones, estos son algunos ejemplos.

Figura 8. Bloques de matemáticas.



Basic number block: Se puede utilizar como cualquier número positivo o negativo (decimales incluidos). Haciendo doble clic en el "0" en el bloque le permitirá cambiar el número.

Subtraction (-): Devuelve el resultado de restar el segundo número de la primera.

Adding (+): Devuelve el resultado de la adición de cualquier cantidad de bloques que tienen un valor numérico junto.

Multiplying (*): Devuelve el resultado de multiplicar cualquier cantidad de bloques que tienen un valor numérico junto.

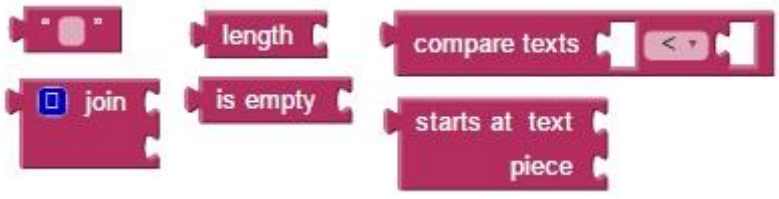
Divide (/): Devuelve el resultado de dividir el primer número por el segundo.

Raised (^): Devuelve el resultado del primera númer elevado a la potencia del segundo número.

1.3.4. Bloques de texto:

Estos son algunos de los bloques de texto.

Figura 9. Bloques de texto



“ “: contiene una cadena de texto.

Join: Agrega todas las entradas para hacer una sola cadena. Si no hay entradas, devuelve una cadena vacía.

Length: Devuelve el número de caracteres, incluyendo espacios en la cadena. Esta es la longitud de la cadena de texto dada.

Is empty: Devuelve si la cadena contiene caracteres (incluyendo espacios). Cuando la longitud de la cadena es 0, devuelve verdadero de lo contrario, devuelve false.

Compare texts: Devoluciones si la primera cadena es lexicográficamente <,> o = la segunda cadena en función de los cuales se selecciona desplegable.

Starts at: Devuelve true si pieza aparece en el texto; de lo contrario, devuelve false.

1.3.5. Lista de bloques

Figura 10. Lista de bloques

Create empty list: crea una lista vacía sin elementos.

Make a list: Crea una lista de los bloques propuestos. Si no se proporciona ningún argumento, esto crea una lista vacía, lo que puede añadir elementos para después.

Is in list?: es uno de los elementos de la lista, devuelve true; de lo contrario, devuelve false

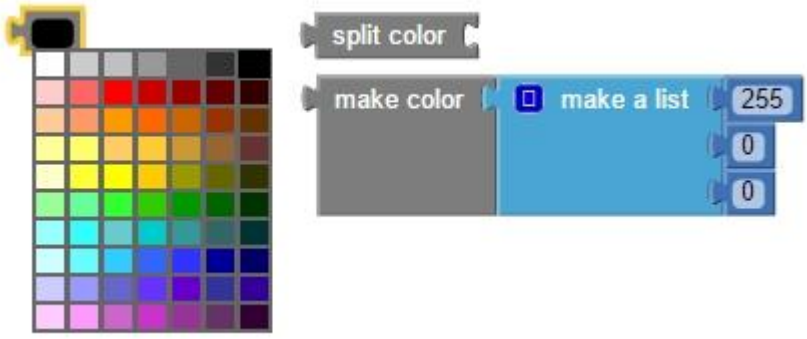
Length of list: devuelve el número de elementos de la lista.

Pick a random ítem list: escoge un elemento al azar de la lista.

Select list ítem list: Selecciona el elemento en el índice que figura en la lista dada.

1.3.6. Colores de bloques:

Figura 11. Colores de bloques.



Basic color blocks: Se trata de un bloque de color básico. Tiene una pequeña forma cuadrada y tiene un color en el medio que representa el color almacenado internamente en este bloque.

Si se hace clic en el color en el centro, una ventana emergente aparece en la pantalla con una tabla de 70 colores que se puede elegir. Al hacer clic en un nuevo color cambiará el color actual del bloque de color básico.

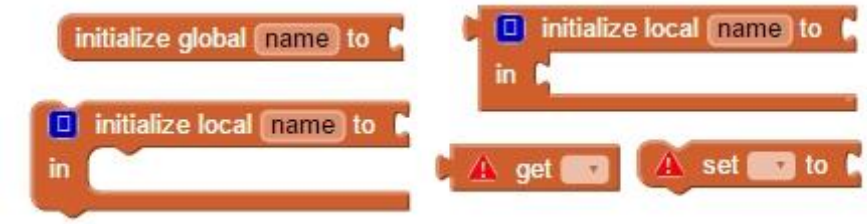
Cada bloque de color básico que se arrastra desde el cajón de colores a la pantalla del editor de bloques mostrará una tabla con los mismos colores cuando se hace clic.

Make color: tiene en una lista de 3 o 4 números. Estos números de esta lista representan valores en un código RGB.

Split color: un bloque de color, variable que contiene un color, o la propiedad de uno de los componentes que representan un color y devuelve una lista de los valores RGB en código RGB de ese color.

1.3.7. Bloques de variables

Figura 12. Bloques de variables



Inialize global: Este bloque se utiliza para crear variables globales. Se necesita, en cualquier tipo de valor como argumento. Al hacer clic en el **name** va a cambiar el nombre de esta variable global.

Las variables globales se pueden cambiar mientras que una aplicación se está ejecutando y pueden ser referidas y cambiadas desde cualquier parte de la aplicación, incluso dentro de los procedimientos y los controladores de eventos.

Get: Este bloque proporciona una manera de conseguir cualquier variable que pueda haber creado.

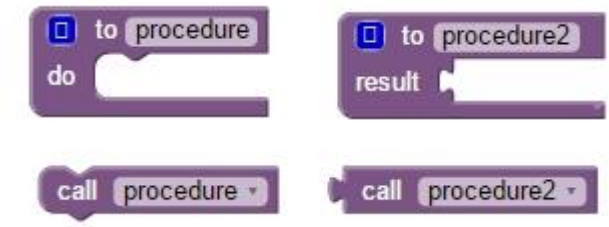
Set to: Este bloque es lo mismo que **get,** solamente que las variables estarán disponibles en la lista desplegable. Una vez que una variable se selecciona, el usuario puede adjuntar un nuevo bloque y darle un nuevo valor.

Inialize local in (do): Este bloque es un mutador que le permite crear nuevas variables que sólo se utilizan en el procedimiento y se ejecutan en la parte **DO** del bloque. De esta manera todas las variables de este procedimiento comenzaran con el mismo valor cada vez que se ejecuta el procedimiento.

Inialize local in (return): Este bloque es un mutador que le permite crear nuevas variables que sólo se utilizan en el procedimiento se ejecuta en la parte **RETURN** del bloque. De esta manera todas las variables en este procedimiento serán todos comenzar con el mismo valor cada vez que se ejecute el procedimiento

1.3.8. Bloques de procedimiento

Figura 13. Bloques de procedimiento



Procedure to: recoge una secuencia de bloques juntos en un grupo. Puede utilizar la secuencia de bloques repetidamente mediante una llamada del procedimiento. Cuando se crea un nuevo bloque de procedimiento, App inventor elige un nombre exclusivo en forma automática.

Call procedure: cuando se crea un procedimiento, app inventor genera automáticamente un bloque de llamadas y lo coloca en el cajón.

Procedure result: igual que “procedure do”, pero si se llama este procedimiento, devuelve el resultado.

Call procedure (plugged): después de la creación de este procedimiento se creará un bloque de llamada que necesita ser enchufado, el resultado de la ejecución de este procedimiento será devuelto en este bloque de llamadas y el valor se transmite al bloque que está enchufado

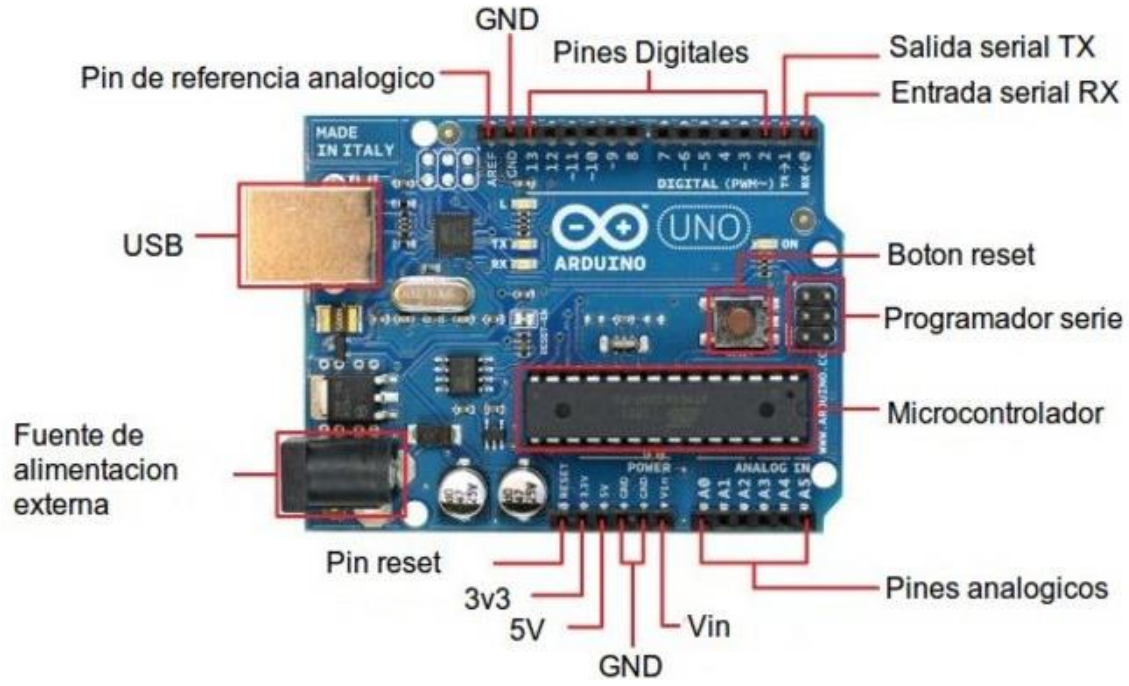
2. PLATAFORMA DE DESARROLLO ARDUINO.

Se describirán las características básicas de la placa Arduino y su software de programación.

2.1. CARACTERÍSTICAS GENERALES DE LA PLACA.

El Arduino Uno (Figura 14), es una placa electrónica basada en el ATmega328 cuenta con 14 pines digitales de entrada / salida (de los cuales 6 se pueden utilizar como salidas PWM), 6 entradas analógicas, un 16 MHz resonador cerámico, una conexión USB, un conector de alimentación, un header ICSP, y un botón de reinicio. Contiene todo lo necesario para apoyar el microcontrolador; simplemente conectarlo a un ordenador con un cable USB o la fuente de poder con un adaptador de CA o la batería a CC para empezar (2).

Figura 14. Placa ARDUINO UNO



2.1.1. Energía

El Arduino Uno puede ser alimentado a través de la conexión USB o con una fuente de alimentación externa. La fuente de alimentación se selecciona automáticamente.

El tablero puede funcionar con un suministro externo de 6 a 20 Volts. Se puede suministra con menos de 7 V, y el pin de 5 V puede suministrar menos de cinco voltios y la junta puede ser inestable. Si se utiliza más de 12 V, el regulador de tensión se puede sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 V (2).

Los pines de alimentación son como sigue:

- **VIN.** La tensión de entrada a la placa Arduino cuando se utiliza una fuente de alimentación externa (por oposición a 5 V de la conexión USB u otra fuente de alimentación regulada).
- **5 V.** Este pin como salida una 5 V regulada del regulador en el tablero. El tablero puede ser alimentado ya sea desde la toma de alimentación de CC (7 - 12 V), el conector USB (5 V), o el pin VIN del tablero (7-12 V).
- **3,3 V.** Un suministro de 3,3 V generada por el regulador a bordo. Corriente máxima es de 50 mA.
- **GND.** Pines de tierra.

2.1.2. Memoria

El ARDUINO UNO R3 tiene 32 KB (con 0,5 KB utilizan para el gestor de arranque). También tiene 2 KB de SRAM y 1 KB de EEPROM.

2.1.3. Entradas y salidas

Cada uno de los 14 pines digitales en el Arduino Uno puede ser utilizado como una entrada o salida, utilizando funciones **pinMode(), digitalWrite(),** y **digitalRead()**. Operan en 5 V. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia de pull-up (desconectado por defecto) de 20 a 50 kOhm. Además, algunos pines tienen funciones especializadas:

- **Serial:** 0 (RX) y 1 (TX). Se utiliza para recibir (RX) y transmitir datos en serie (TX) TTL. Estos pines están conectados a los pines correspondientes del ATmega8U2 USB a TTL chip de serie.
- **Interrupciones externas:** 2 y 3. Estos pines pueden configurarse para activar una interrupción en un valor bajo, un flanco ascendente o descendente, o un cambio en el valor.
- **PWM:** Corresponden a los pines 3, 5, 6, 9, 10, y 11, proporcionan una salida PWM de 8 bits con la función **analogWrite()**.
- **SPI:** Corresponden a los pines 10 (SS), 11 (MOSI), 12 (MISO) y 13 (SCK) estos pines admiten la comunicación SPI. Serial Peripheral Interface (SPI) es un protocolo de datos en serie síncrono utilizado por los microcontroladores para comunicarse con uno o más dispositivos periféricos rápidamente en

distancias cortas. También se puede utilizar para la comunicación entre dos microcontroladores.

- **LED:** 13. Hay un LED incorporado conectado al pin digital 13. Cuando el pin es de alto valor, el LED está encendido, cuando el pin es bajo, está apagado.

El ARDUINO UNO tiene 6 entradas analógicas, etiquetadas desde A0 hasta A5, cada una de las cuales proporcionan 10 bits de resolución (es decir, 1024 valores diferentes).

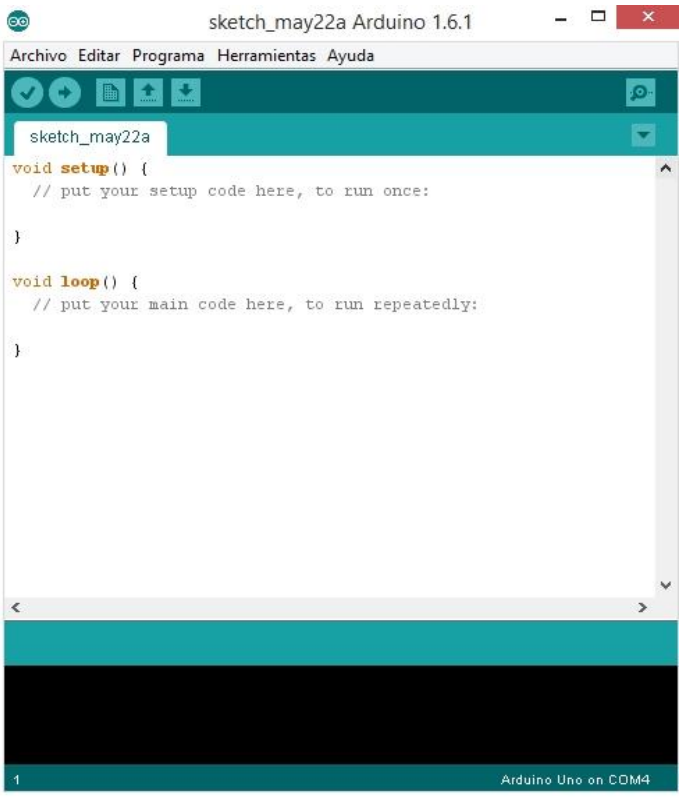
2.1.4. Características físicas.

La longitud máxima y la anchura del ARDUINO UNO son 2,7 pulgadas (6,85 cm) y 2,1 pulgadas (5,33 cm), respectivamente, con el jack conector USB y el poder que se extiende más allá de la dimensión anterior. Cuatro orificios de los tornillos permiten la junta para fijarse a una superficie o caso.

2.2. SOFTWARE DE PROGRAMACIÓN EN ARDUINO 1.6.1.

El software de Arduino 1.6.1 es el entorno de desarrollo para programar la placa, en la
Figura 15 se muestra el aspecto del entorno de programación.

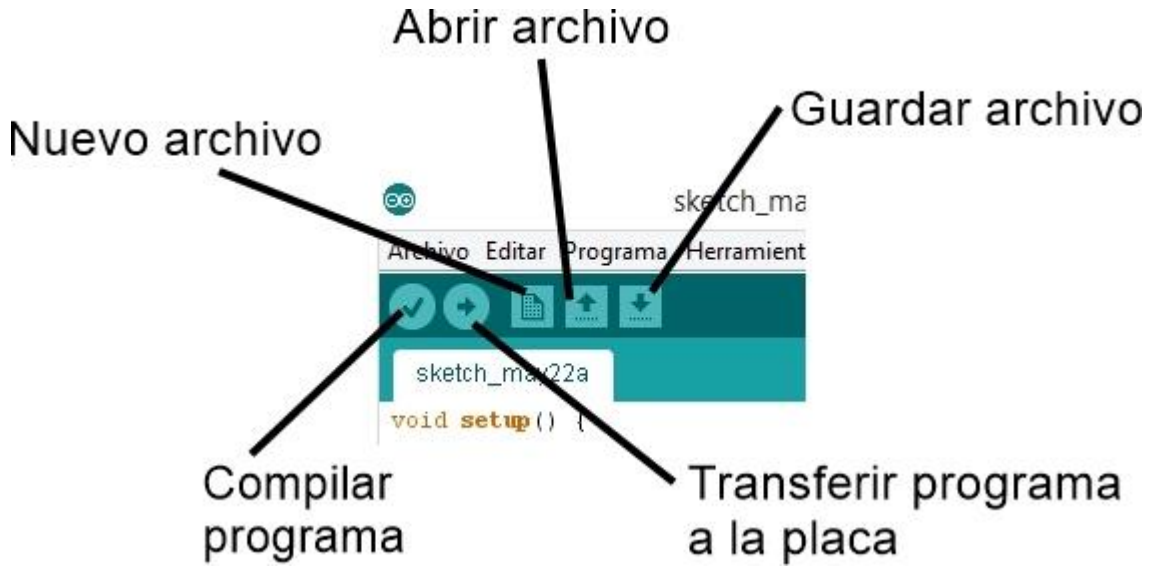
Figura 15. Software de programación de Arduino.



Para comenzar a trabajar con el entorno de desarrollo de Arduino es necesario configurar las comunicaciones entre la placa Arduino y el PC. Para ello se debe abrir en el menú “Tools” la opción “serial port”. En esta opción se debe seleccionar el puerto serie al que está conectada la placa. Si desconocemos el puerto al que está conectado la placa, se puede hallar a través del administrador de dispositivos (puertos COM & LPT/ USB serial port).

En la **¡Error! La autoreferencia al marcador no es válida.** se encuentran los botones básicos para compilar, transferir el programa, abrir y guardar el archivo.

Figura 16. Descripción botones



2.2.1. Estructura básica de un programaEn la **¡Error! La autoreferencia al marcador no es válida.** se encuentran los botones básicos para compilar, transferir el programa, abrir y guardar el archivo.

Figura 16. Descripción botones

La estructura básica de programación de Arduino es bastante simple y divide la ejecución en dos partes: **loop()** y **Setup()** constituye la parte inicial del programa y **loop()** es la ejecución. En la función **setup()** se incluye la declaración de variables y se trata de la primera función que se ejecuta en el programa. Esta función se ejecuta una única vez y es empleada para configurar el **pinMode** e iniciar la comunicación serie. La función **loop()** incluye el código a ser ejecutado continuamente.

Se puede observar en el Cuadro 1 un bloque de código, cada instrucción finaliza con punto y coma ";" y los comentarios se indican con "//". Al igual que en C se pueden introducir bloques de comentarios.
Cuadro 1. Estructura básica de un código en Arduino.

```
void setup() {
  // Put your setup code here, to run once:

 PinMode (pin, OUTPUT);      // Establece 'pin' como salida
 }

 Void loop () {
  // put your main code here, to run repeatedly:
```

```
DigitalWrite (pin, HIGH);      //Activa 'pin' como salida
Delay (1000);                  // Pausa un segundo
DigitalWrite (pin, LOW);       // desactiva 'pin'
Delay (1000);
}
```

2.2.2. Funciones de Arduino versión 1.6.1.

Una función es un bloque de código identificado por un nombre y que es ejecutado cuando la función es llamada. Ver Cuadro 2. Ejemplo de una función.Cuadro 2.

Cuadro 2. Ejemplo de una función.

```
Int delayVal () {

 Int v;                        //crea una variable temporal 'v'
 v = analogread(pot);          // lee el valor del potenciómetro
 v /=4                         // convierte los valores 0-1023 a 0-255

 return v;                     // devuelve el valor final de la variable
 }
```

2.2.3. Variables

Una variable debe ser declarada con un determinado valor. En la declaración de la variable se indica el tipo de datos que almacenará (**int, float, long**)

Int inputVariable = 0;

Una variable puede ser declarada en el inicio del programa antes de **setup(),** localmente a una determinada función e incluso dentro de un bloque como puede ser un bucle. Una variable puede ser global o local, una variable global es aquella que puede ser empleada en cualquier función del programa y una variable local es la que se le otorga un ámbito local. Tales variables sólo pueden accederse desde la función o bloque de instrucciones en donde se declaran. Las variables locales se contraponen a las variables globales. Estas variables deben ser declaradas al inicio del programa ver el Cuadro 3.

Cuadro 3. Ejemplo de una variable.

```
Int v;                          // 'v' es visible en todo el programa
void setup() {

}
```

```
void loop ()   {
for (int i=0;  i<20;)                  // ‘i’ es visible solo en el bucle

i++;
float f;                               // ‘f’ es visible únicamente en la función loop()
}
```

2.2.4. Operadores aritméticos

Empleando variables, valores constantes o componentes, pueden realizarse operaciones aritméticas. Además pueden hacerse las siguientes asignaciones como por ejemplo:

Cuadro 4. Operadores aritméticos

x++. Lo mismo que x = x +1.
x--. Lo mismo que x=x-1.
x += y. Lo mismo que x=x+y.
x -= y. Lo mismo que x=x-y.
x *=y. Lo mismo que x=x*y.
x /= y. Lo mismo que x=x/y.

El lenguaje de Arduino presenta las siguientes constantes predefinidas:

- TRUE / FALSE
- HIGH/LOW. Estas constantes definen los niveles de los pines como: HIGH se define como el nivel lógico 1 (ON) o 5 V. Y LOW es el nivel lógico 0 (OFF), o 0 V.
- INPUT/OUTPUT. Constantes empleadas con la función pinMode() para definir el tipo de un pin digital que puede ser usado como entrada INPUT o como salida OUTPUT. Ejemplo pinMode(13, OUTPUT);

2.2.5. Sentencias condicionales.

El lenguaje de Arduino permite realizar sentencias condiciones if, if-else, for, while, do-while. Su utilización es similar a las funciones en lenguaje C.

2.2.6. Entradas y salidas digitales y analógicas

- **Función pinMode (pin, mode):** Es usada en la función **setup()** para configurar un pin dado para comportarse como INPUT o OUTPUT. Los pines de Arduino funcionan como entradas.

- **Función digitalRead(pin):** Introduce un nivel alto (HIGH) o bajo (LOW) en el pin digital específico, el pin puede ser especificado con una variable o una constante 0-13. Ejemplo. DigitalWrite (pin, HIGH).
- **Función analogWrite(pin, value):** Escribe un valor pseudo-analógico usando modulación por ancho de pulso (PWM) en un pin de salida marcado como PWM.

2.2.7. Funciones de tiempo y matemáticas.

- **delay (ms):** realiza una pausa en el programa, la cantidad de tiempo en milisegundos especificada en el parámetro (máximo 1000, mínimo 1).

- **millis ():** Devuelve la cantidad de milisegundos que lleva la placa Arduino. Ejecutando el programa actual.

- **min(x, y) y max(x, y):** devuelven el mínimo y el máximo respectivamente entre sus parámetros.

2.2.8. Funciones de generación aleatoria.

- **randomSeed (seed):** Es un valor o semilla como el punto de inicio para la función **random().** Este parámetro debe ser realmente aleatorio y para ello se puede usar la función **millis().**
- **random (max), random(min, max):** esta funcion devulve un valor aleatorio entre el rango especifico.

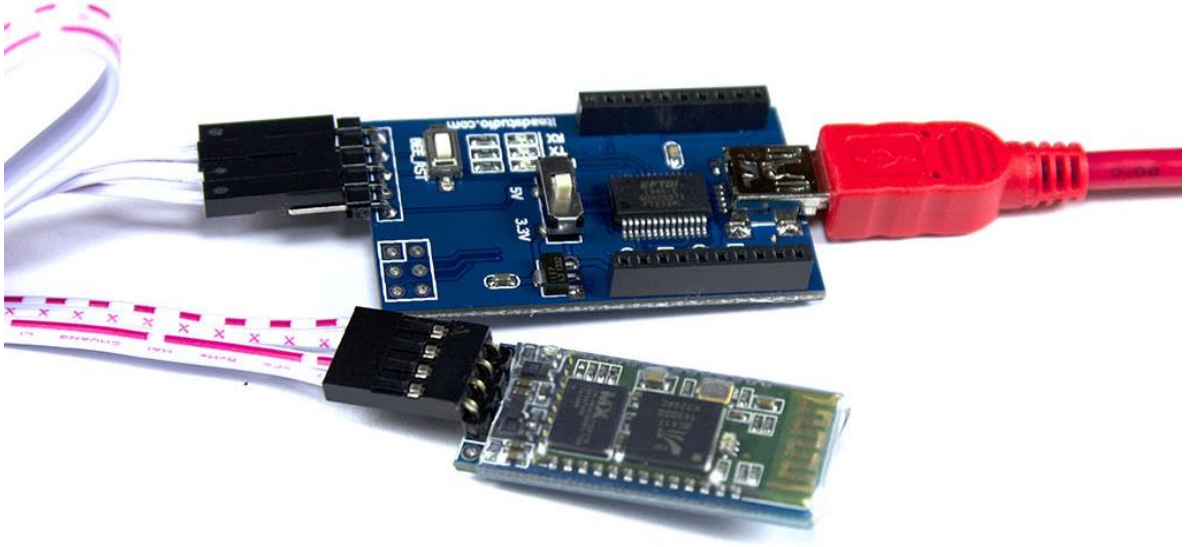
2.3. INTERFAZ DE CONFIGURACIÓN DE COMANDOS AT EN HC-06

Los comandos AT son instrucciones codificadas que conforman un lenguaje de comunicación entre el hombre y un terminal modem.

El HC-06 tiene un firmware distinto y también un funcionamiento distinto en cuanto a su modo de configuración. Para poder configurar el HC-06 es necesario que este NO esté emparejado ni siendo usado por ningún dispositivo. De igual forma que el HC-05 es necesario conectarlo a la PC y usar un programa de terminal para darle

instrucciones de configuración (Comandos AT), aunque también se puede escribir un programa de Arduino o en un microcontrolador para configurarlo. Para conectarlo con la PC se utiliza un adaptador USB serial como se muestra en la Figura 17.

Figura 17. Realizar pruebas con el módulo HC-06 y una tarjeta USB serial con FT-232RL



El módulo HC-06 acepta un set muy básico de comandos, que permite pocas configuraciones, pero que sin duda será útil para personalizar este económico módulo y configurarlo para satisfacer las necesidades de la aplicación.

Los comandos son:

- Prueba de funcionamiento:
  Enviar: AT
  Recibe: OK
- Configurar el Baudrate:
  Enviar: AT+BAUD<Numero>
  El parámetro **número** es un carácter hexadecimal de '1' a 'c' que corresponden a los siguientes Baud Rates: 1=1200, 2=2400, 3=4800, 4=9600, 5=19200, 6=38400, 7=57600, 8=115200, 9=230400, A=460800, B=921600, C=1382400
  Recibe: OK<baudrate>
- Configurar el Nombre de dispositivo Bluetooth:
  Enviar: AT+NAME<Nombre>
  Recibe: OKsetname
- Configurar el código PIN de emparejamiento:
  Enviar: AT+PIN<pin de 4 dígitos>

Recibe: OK<pin de 4 dígitos>

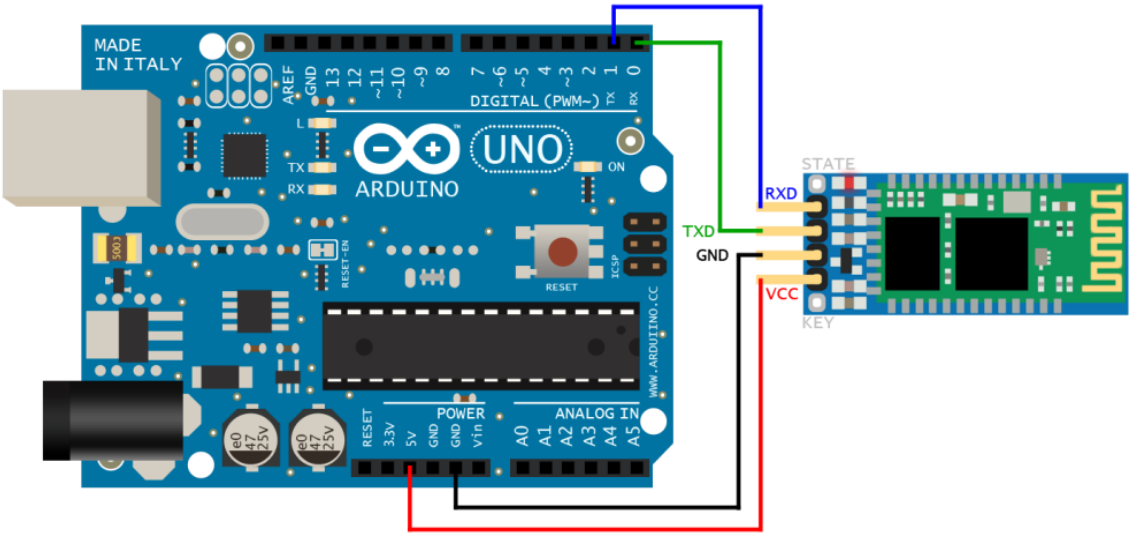
- Obtener la versión del firmware:
  Enviar: AT+VERSION
  Recibe: Linvor1.8

Se debe tener en cuenta de introducir todas las letras del comando en mayúsculas ya que de lo contrario no funcionarán.

2.4. CONEXIÓN BÁSICA CON ARDUINO UNO R3

Las conexiones para realizar con Arduino uno R3 son bastante sencillas, solamente se requiere colocar como mínimo la alimentación y conectar los pines de transmisión y recepción serial (TX y RX). Hay que recordar que en este caso los pines se deben conectar cruzados TX Bluetooth -> RX de Arduino y RX Bluetooth -> TX de Arduino. Ver Figura 18, muestra las conexiones básicas para que funcione el módulo.

Figura 18. Conexión entre el modulo Bluetooth HC-06 y el módulo Arduino uno R3



Para lograr la conexión y que se pueda cargar el código al módulo Arduino uno R3, se usó una conexión por medio de un cable USB.

3. DISEÑO DE LA APLICACION PARA EL CONTROL DE ILUMINACION CON APP INVENTOR.

3.1. SE CREO UNA APP PARA EL CONTROL DE ILUMINACION A DISTANCIA POR MEDIO DE CONEXIÓN BLUETOOTH

En este proyecto se realizó una aplicación Android con App Inventor para el control de iluminación a distancia, estás iluminarias estan conectadas a un módulo Arduino uno R3 comunicados a través de Bluetooth. Se construyó la aplicación con App inventor. Es explicada paso a paso su proceso.

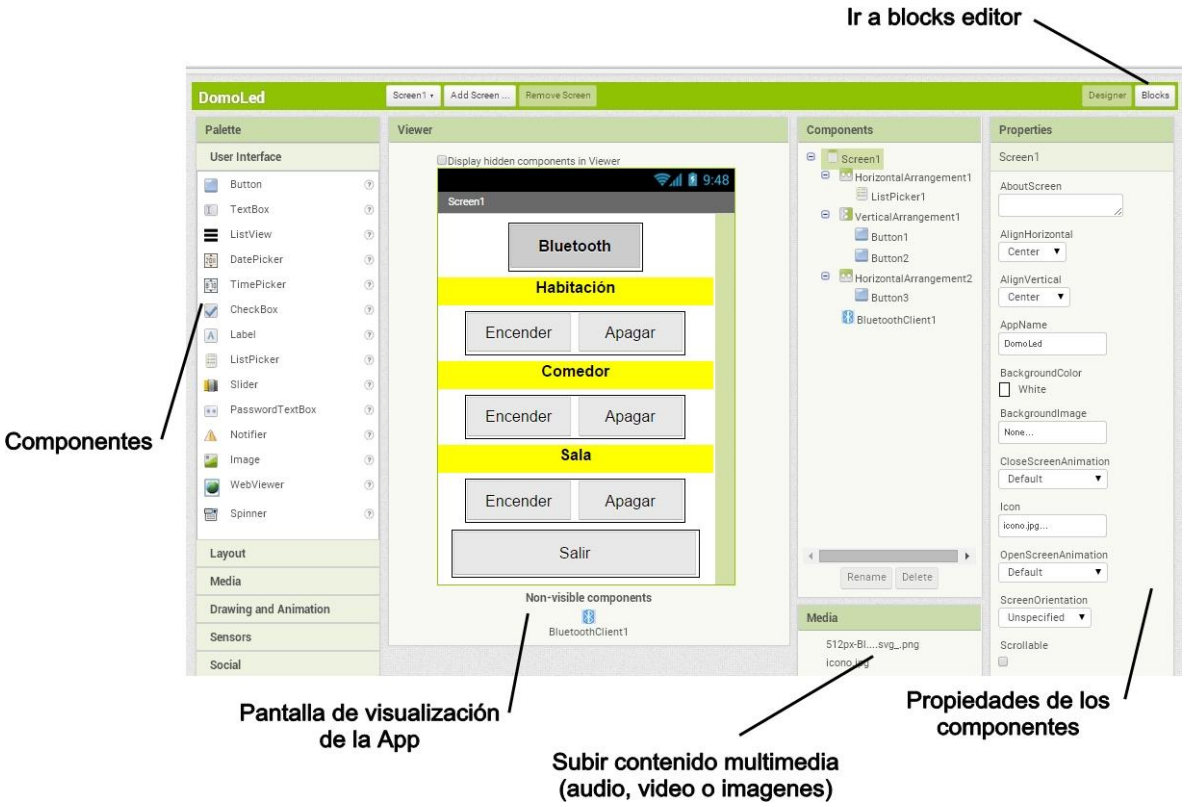
3.2. APP INVENTOR

Después de registrarse se crea un nuevo proyecto en App inventor como se explicó en la sección 1.1. GESTOR DE PROYECTOS, después se construye el diseño de la App. En el centro se tiene la visualización de una pantalla de un dispositivo Android donde se irán poniendo todos los componentes que se quiera usar. Estos componentes se encuentran en la parte izquierda y se tiene desde botones, etiquetas, sliders, etc. También se puede encontrar el hardware del dispositivo Android como la cámara, GPS, acelerómetro, sonido, etc. Esto es muy útil ya que se tiene la posibilidad de usar sensores del dispositivo Android junto a Arduino para el proyecto.

Se hizo el diseño de la aplicación o App tal como se describió en la sección 1.2. DISEÑADOR, el resultado se puede observar en la Figura 19.

En esta app se ha puesto un botón que servirá para conectarse al Arduino a través de Bluetooth, 6 botones para el encendido y el apagado de la luminaria y un botón de salida para detener la conexión vía Bluetooth con el módulo Arduino; se pondrá un cliente Bluetooth para establecer la comunicación Bluetooth, también se ha puesto una imagen icono de la app subiendo una imagen en **Media** y colocando el archivo en “Icon” de las propiedades de la pantalla (screen). Las demás propiedades de los componentes sirven para cambiar al gusto de cualquier persona los colores, medidas y nombres de estos. Ver la Figura 19.

Figura 19. Diseño exterior de la aplicación.

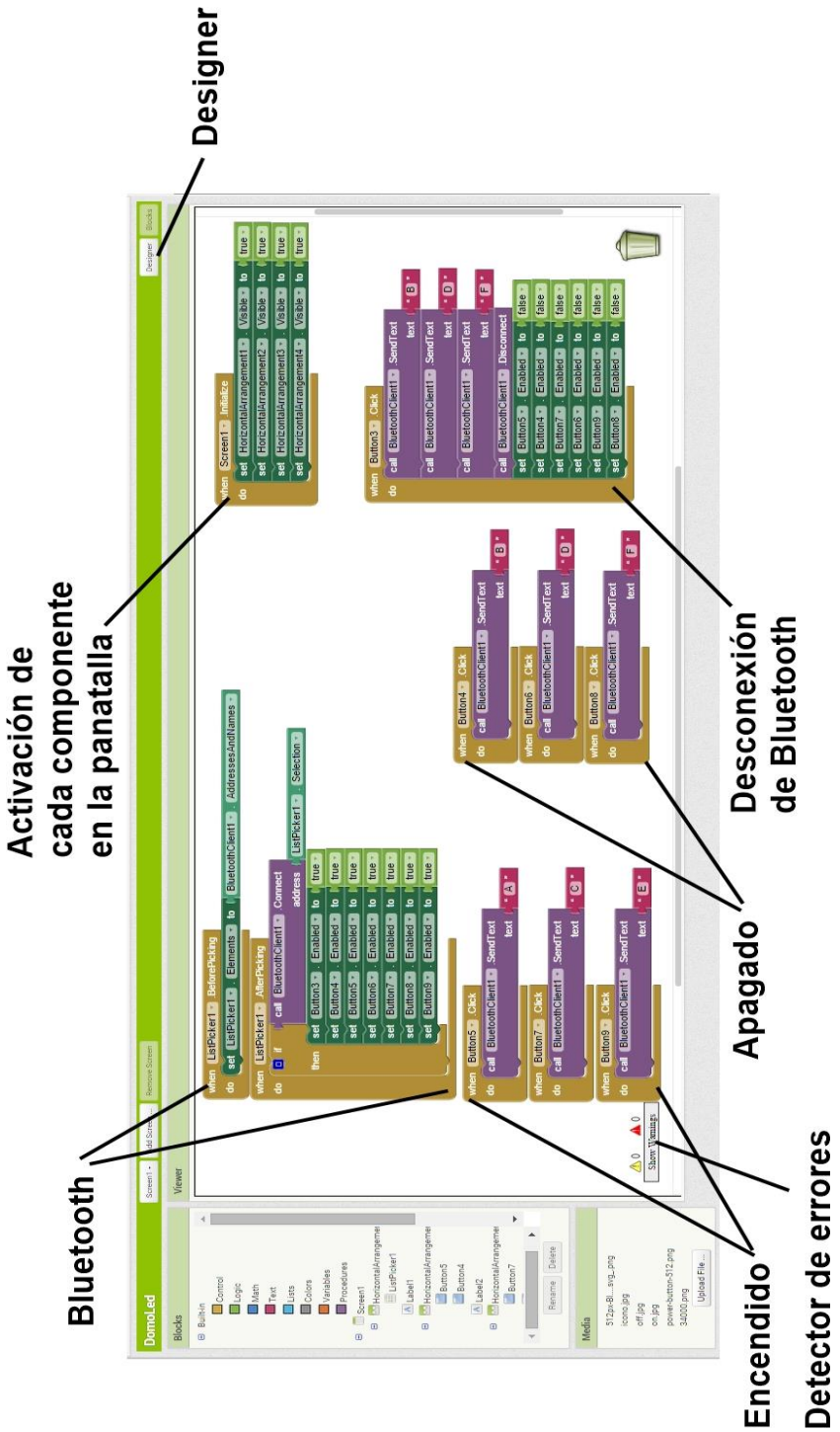


Una vez terminado con el diseño de la app, se hace click en la el botón Blocks en la parte superior derecha de la pantalla para ir al App Inventor Blocks Editor. Aquí es donde se programa la app en forma de bloques como si se tratase de un rompecabezas.

3.3. DISEÑO DEL PROGRAMA INTERNO CON BLOCKS EDITOR.

En **Viewer** se tiene una hoja en blanco donde se ubicarán los bloques, tal como se explicó en la sección 1.3. EDITOR DE BLOQUES, estos bloques están en la parte izquierda en el apartado Blocks donde se encontraran los elementos y las funciones más comunes en los lenguajes de programación tales como sentencias, funciones lógicas, funciones matemáticas, **strings**. También estarán los elementos y las funciones específicas de los componentes que se insertaron en la aplicacion, que en este caso son Bluetooth, y 7 botones ver la Figura 20.

Figura 20. Diagrama de bloques.



Bluetooth: La comunicación Bluetooth depende del botón llamado **Listpicker1** que se puso. Antes de pulsarlo, si el Bluetooth del dispositivo Android está listo, la app recogerá todas las conexiones Bluetooth listadas en este. Esto lo hará al principio de abrir la app. Una vez se pulse el botón **CONECT** saldrá un listado donde se selecciona el módulo Bluetooth conectado al Arduino. Ver la Figura 21.

Botones: Para la parte de los botones de encendido (Buttom5, Buttom7 y Buttom9) y apagado (Buttom4, Buttom6 y Buttom8), los botones tienen capacidad de detectar clics. Muchos aspectos se pueden cambiar como, la imagen que se le asigna a los botones de encendido y apagado, así como se puede hacer clic, se puede cambiar en el Diseñador o en el Editor de bloques, en este proyecto no se le asigno imágenes para los botones de encendido o apagado. Ver la Figura 21.

Esta aplicación está diseñada solo para controlar 3 luminarias o 3 ramales; este proyecto se realizó para contralar 3 lugares diferentes de una vivienda, si el usuario lo desea se puede programar la aplicación para que controle múltiples luminarias con un solo modulo Bluetooth.

Figura 21. Diseño de la aplicación

4. DISEÑO DE CIRCUITOS ELECTRICOS Y ELECTRONICOS CON LA PLATAFORMA ARDUINO.

Cuando se envían los valores de los botones Button1 o Button2 mediante vía Bluetooth hacia el Arduino, se envía una variable “A” en el caso de encendido, y “B” en el caso de apagado, se uso la función *serial.read()* con esta función se podrá leer los datos o variables que lleguen desde la aplicación, en este caso, si recibe una “A” Arduino encenderá la iluminaria conectada, y recibe una “B” Arduino apagara la luminaria.

Como se muestra en el **¡Error! La autoreferencia al marcador no es válida.**, depende de la variable que reciba el módulo Arduino, si el dato recibido es igual “A” Arduino envía un “1” al pin 4 y se activa la luminaria, si el dato recibido es igual a “B” entonces el módulo Arduino envía un 0 al pin “4”, y asi con los otros 2 botones se trabaja el mismo procedimiento, para apagar la luminaria; el encendido y el apagado de la luminaria tienen un retardo de un segundo.

Cuadro 5. Código completo

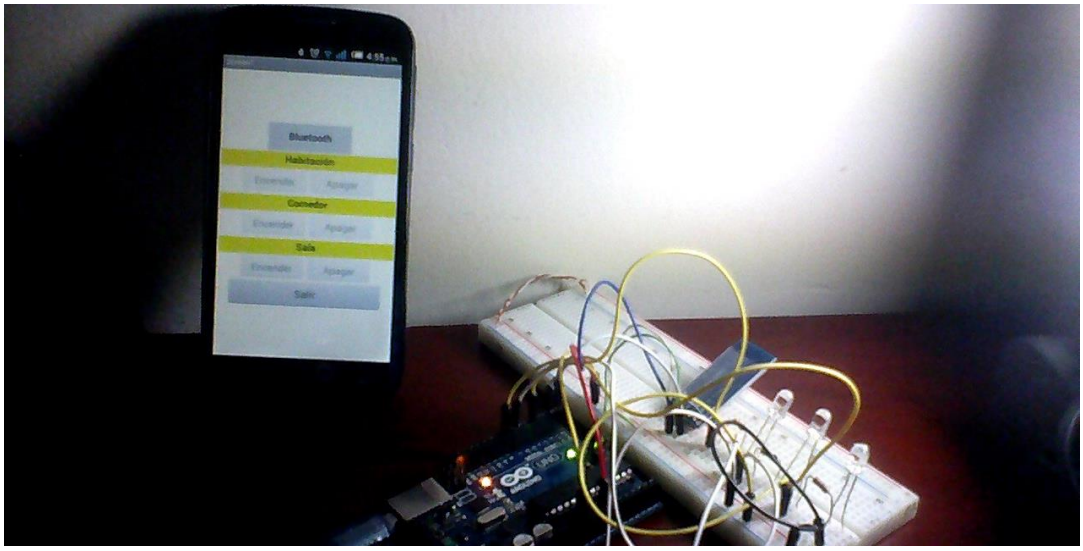
```
void setup()
{
Serial.begin(9600);
pinMode(4, OUTPUT); // pin de salida
pinMode(7, OUTPUT); // pin de salida
pinMode(8, OUTPUT); // pin de salida
}

void loop(){

char dato=Serial.read(); //leer la variable enviada desde la aplicación

if(dato == 'A') // si el dato recibido es igual a A
{
 digitalWrite(4,HIGH);// manda un 1 al pin 4
}
if(dato == 'B')// si el dato recibido es igual a B
{
  digitalWrite(4,LOW); // manda un 0 al pin 4
}

if(dato == 'C') // si el dato recibido es igual a C
{
 digitalWrite(7,HIGH);// manda un 1 al pin 7
}
if(dato == 'D')// si el dato recibido es igual a D
{
  digitalWrite(7,LOW); // manda un 0 al pin 7
}

if(dato == 'E') // si el dato recibido es igual a E
{
 digitalWrite(8,HIGH);// manda un 1 al pin 8
}
if(dato == 'F')// si el dato recibido es igual a F
{
  digitalWrite(8,LOW); // manda un 0 al pin 8
}
delay(1000); //retardo de un segundo
}
```

5. PRUEBAS Y RESULTADOS

5.1. CONSTRUCCIÓN DEL CIRCUITO ELÉCTRICO

Esta primera prueba se realizó con éxito un control de 3 leds por separado.

Figura 22. Circuito eléctrico

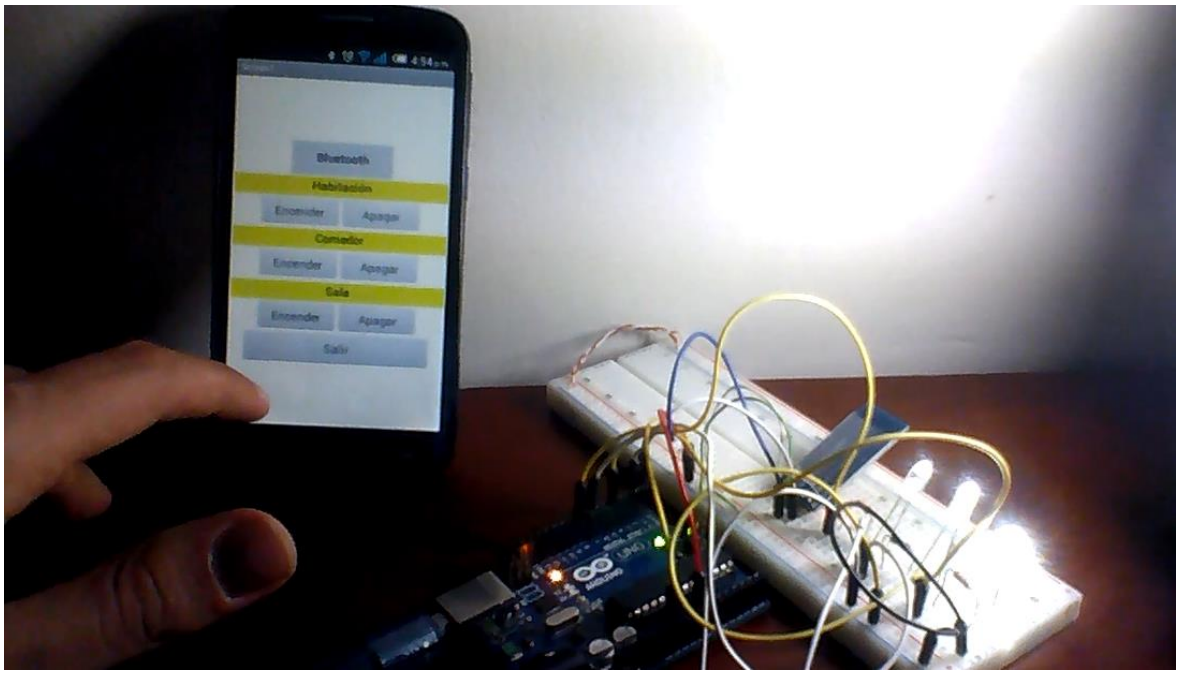


Como se puede observar en la

Figura 23. Circuito eléctrico funcionando

Figura 23 el circuito eléctrico enciende correctamente

Figura 23. Circuito eléctrico funcionando

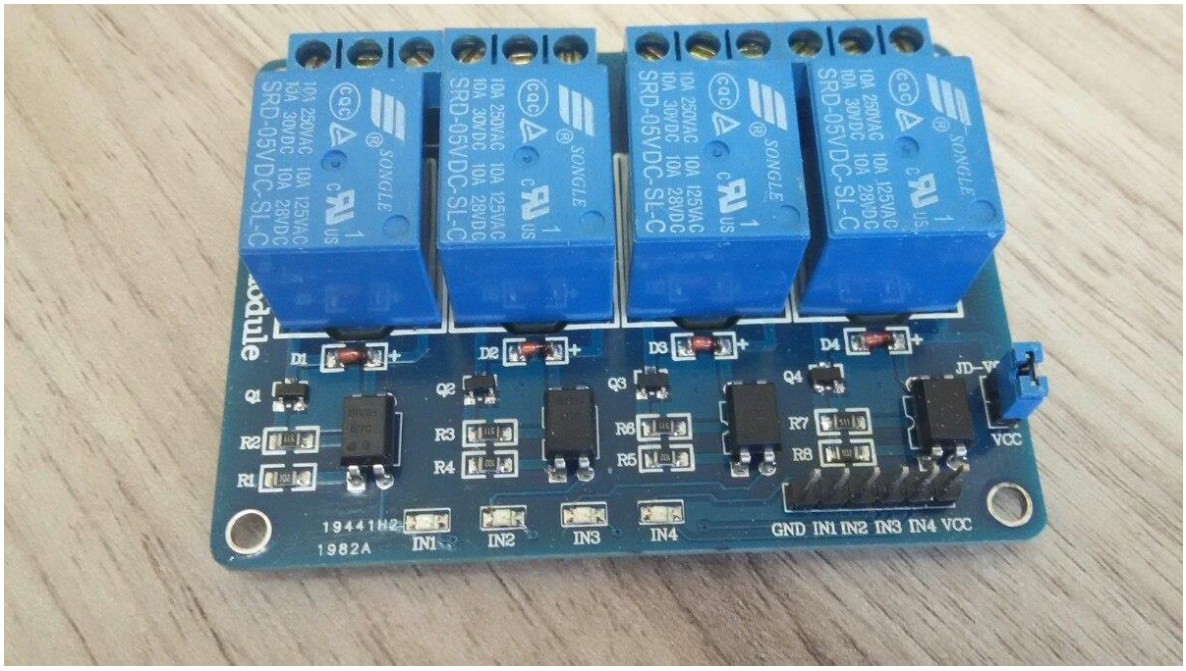


5.2. CIRCUITO DE POTENCIA.

Para el circuito de potencia de este proyecto se usó una placa de 4 relés para Arduino, esta placa es usualmente usada para controlar el encendido y apagada de luces, controlar el riego del jardín, etc. Esta fue la mejor opción de este proyecto. Ver Figura 24.

La placa de relés Arduino es capaz de manejar cargas de alta tensión gracias a sus 4 canales independientes con relés electromagnéticos protegiendo el circuito de control por opto acoplador. La potencia máxima de conmutación 300 W AC o DC.

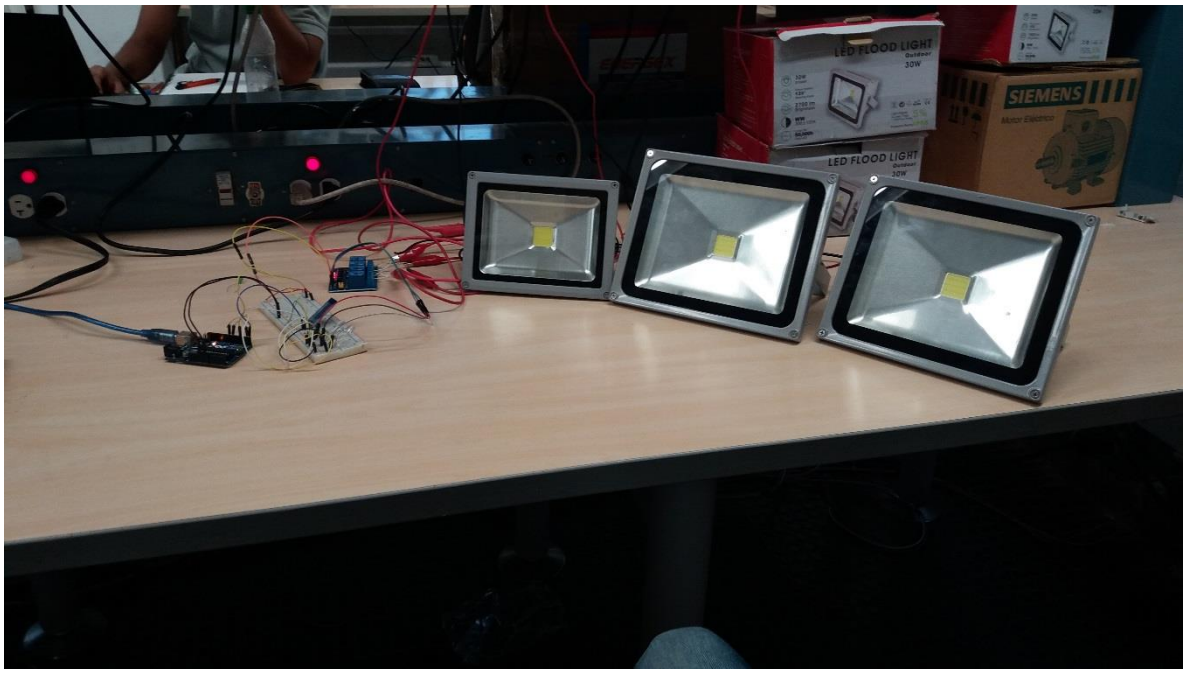
Figura 24. Placa de 4 relés para Arduino.



Para el resultado final se hizo el acople de todo el sistema y se controló con la aplicación diseñada para la activación o desactivación de las luces.

Figura 25 Luminaria tipo led, led flood light 30W

Figura 26. Resultado final



6. CONCLUSIONES

El sistema eléctrico para el control de la iluminación a distancia de hogares, escuelas y sitios sociales como bares o restaurantes fue realizado exitosamente con el uso de tecnologías como Bluetooth y Android se logró controlar tres luminarias de forma independiente a una distancia máxima de 10 metros.

Se concluyó que no solo se pueden controlar tres luminarias, sino que también se puede controlar más luminarias de forma independiente o circuitos ramales diferentes solo con hacer unos pequeños ajustes en el programa de Android y el circuito de potencia; la cantidad de bombillas dependerá de la potencia del relevador que se utilice.

La implementación de sistemas de domótica eficientes y de bajo costo es posible y se puede brindar a los usuarios con limitaciones físicas una mejor calidad de vida, desde el punto de vista de confort.

Este dispositivo fue diseñado para una conexión inalámbrica sim embargo también se podría agregar una conexión manual.

Este proyecto fue muy enriquecedor en cuanto a la adquisición de nuevos conocimientos como lo fue Appinventor y Arduino, ya que se trabajó en áreas como la electrónica y la programación de dispositivos hardware; este proyecto deja una base de nuevos conocimientos acerca de Android y Arduino para los estudiantes que quieran mejorar este proyecto.

BIBLIOGRAFÍA

1. Android <<www.android.com>> Disponible en: <https://developer.android.com/guide/index.html>. [En línea]

2. Arduino, <<www.arduino.cc>> Disponible en: <http://www.arduino.cc/en/Main/ArduinoBoardUno>. [En línea]

3. ROMÁN JIMÉNEZ, Roger Alonso. *Diseño de un sistema domótico para control de iluminación y monitoreo de consumo eléctico. Trabajo de grado Diseñador Industrial. Bucaramanga. Universidad Industrial de Santander. Facultad de Ingenierías Fisico - Mecánicas. 2011. 186 p. .*

4. MONTOYA ÁLVAREZ, David. SUÁREZ, Francisco Javier y ORTIZ, Luisa Fernanda. *Juego de ajedrez sobre dispositivos móviles con sistema operativo android para personas con limitación visual. Trabajo de grado Ingeniería de Sistemas y Computación. Pereira Risaralda. Universidad Tecnológica de Pereira. Facultad de Ingenierías. 2014.*

5. ANDRADE FERNANDEZ, Alejandro. *Implementación del sistema de domótica en el hogar. Trabajo de grado Ingeniero de Sistemas y Telecomunicaciones. Pereira, Risaralda. Universidad Católica de Pereira. Facultad de Ciencias Básicas en Ingeniería, 2013. 77 p.*

6. LUGO MAYORGA, Francisco Armando. *Robot controlado inalámbricamente por un dispositivo Android. Trabajo de grado Ingeniero Mecatrónica. México. Universidad Nacional Autónoma de México. Facultad de Ingeniería. 2014. 110 p.*

7. MOLINA RIVERA, Yeicy Juliana. SANDOVAL CARDONA, Jonathan y TOLEDO FRANCO, Santiago Alberto. *Sistema operativo Android: características y funcionalidad para dispositivos móviles. Trabajo de grado Ingeniería de Sistemas y Computación. Pereira Risaralda. Universidad Tecnológica de Pereira. Facultad de Ingenierías. 2012.*

8. PATIÑO, Anderson Alfonso. *Diseño y elaboración de la guía para sistemas digitales con Arduino uno R3. Trabajo de grado Tecnólogo en Electricidad. Pereira. Universidad Tecnológica de Pereira. Facultad de Tecnología. 2014.*

9. MORA PATIÑO, Jorge Eduardo y SALAZAR TABARES, Diego Fernando. *Diseño e implementación de sistema de propulsión y control para silla de ruedas. Trabajo de grado Ingeniero en Mecatrónica. Pereira. Facultad de Tecnologías. 2014.*

10. MONTOYA ÁLVAREZ, David. SUÁREZ, Francisco Javier y ORTIZ, Luisa Fernanda. *Juego de ajedrez sobre dispositivos móviles con sistema operativo android para personas con limitación visual. Trabajo de grado Ingeniería de*

Sistemas y Computación. Pereira Risaralda. Universidad Tecnológica de Pereira. Facultad de Ingenierías. 2014. .

11. App Inventor <<appinventor.mit.edu/explore>> Disponible en: <http://appinventor.mit.edu/explore/ai2/support/blocks.html>. [En línea]

12. ANDRADE FERNANDEZ, Alejandro. *implementacion del sistema de domotica en el hogar. Trabajo de grado ingeniero de sistemas y telecomnicaciones. Pereira Risaralda.Universidad Catolica de Pereira. Facultad de ciencias basicas en ingieneria, 2013. 77 p.*

13. POMARES BAEZA, Jorge. *Manual de Arduino. Grupo de innovación Educativa en Automática Universidad de Alicante. 2009. 9 p.*

14. Wikipedia <<es.wikipedia.org>> Disponible en: <es.wikipedia.org/wiki/Domótica>. [En línea]