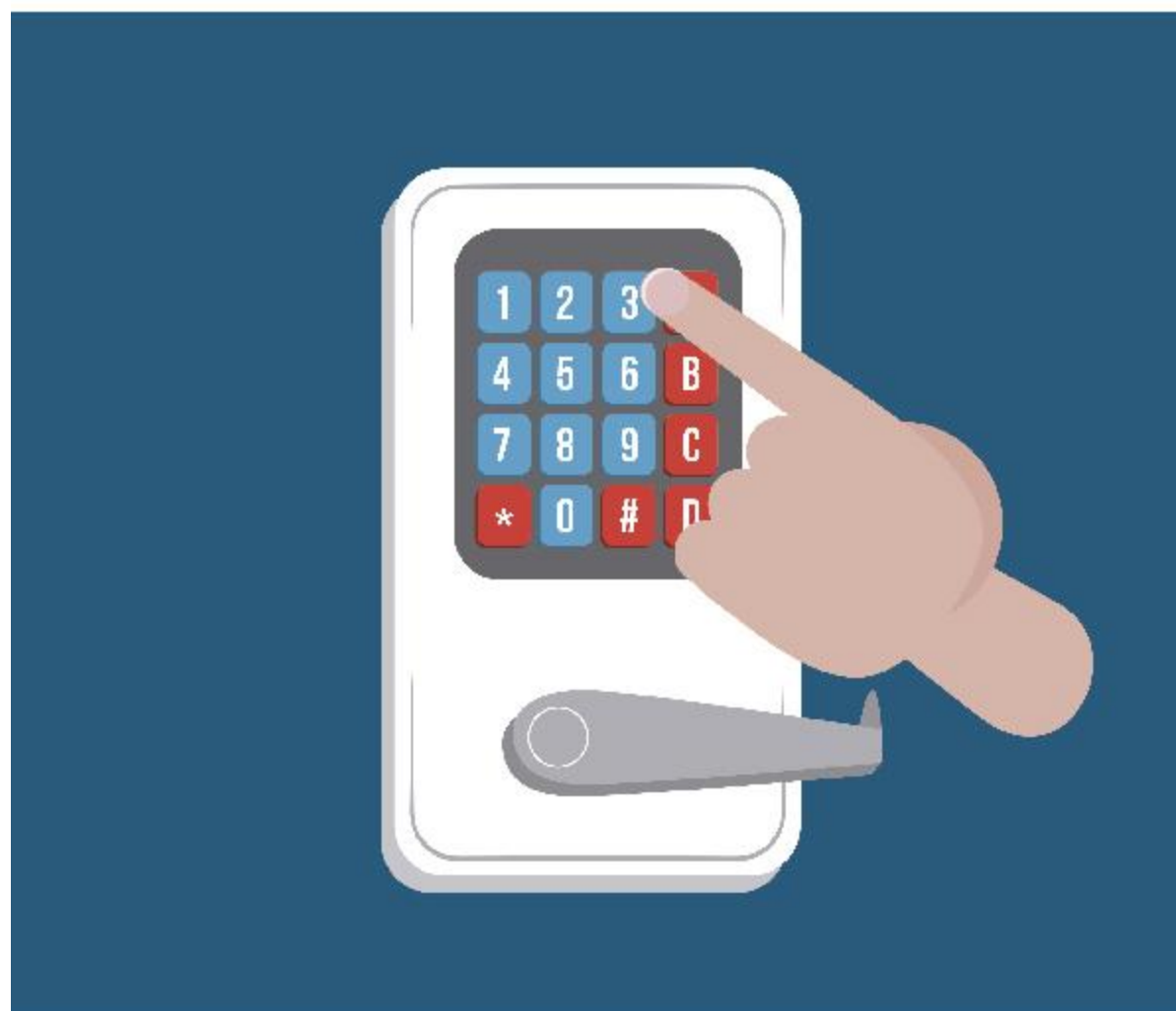


SABERES DIGITALES



CERRADURA AUTOMATIZADA DOMICILIARIA

AUTORIDADES

Presidente de la Nación

Mauricio Macri

Vicepresidenta de la Nación

Marta Gabriela Michetti

Jefe de Gabinete de Ministros

Marcos Peña

Ministro de Educación, Cultura, Ciencia y Tecnología

Alejandro Finocchiaro

**Titular de la Unidad de Coordinación General del
Ministerio de Educación, Cultura, Ciencia y Tecnología**

Manuel Vidal

Subsecretario de Coordinación Administrativa

Javier Mezzamico

Director Ejecutivo INET

Leandro Goroyesky

Gerenta General de EDUCAR Sociedad del Estado

Liliana Casaleggio

Directora Nacional de Asuntos Federales

María José Licio Rinaldi

Director Nacional de Educación Técnico - Profesional

Fabián Prieto

Coordinador de Secundaria Técnica

Alejandro Anchava

Responsable de Formación Docente Inicial y Continua INET

Judit Schneider

Coordinador General En FoCo

Pablo Trangone

AUTORIDADES	¡Error! Marcador no definido.
CERRADURA AUTOMÁTICA	4
Ficha técnica	4
Presentación	5
Desarrollo	6
Nivel Inicial	6
Paso 1 - Realizar el montaje del servomotor en una Protoboard	6
Paso 3 - Subir el código a la placa Arduino	11
Paso 4 - Conectar los LED indicadores	12
Paso 5 - Programar los LED indicadores	12
Paso 6 - Comenzar a armar la maqueta	14
Paso 7 - Imprimir las piezas	14
Paso 8 - Ensamblar el sistema mecánico con el servomotor	15
Nivel Intermedio	16
Paso 1 - Subir el código a través de Arduino IDE.	17
Paso 2 - Incluir librería “Keypad”	20
Paso 3 - Programando el teclado de membrana	21
Nivel Avanzado	25
Paso 1 - Introducción a Internet de las Cosas (IoT)	25
Paso 2 - Crear un Panel de Control	26
Paso 3 - Conectar módulo Obloq	32
Paso 4 - Arduino IDE	33
Paso 6 - Programación IoT	34
Cierre	40
Reconocimientos	¡Error! Marcador no definido.

CERRADURA AUTOMÁTICA

Ficha técnica

Nivel educativo	Secundario. Ciclo Básico.
Descripción general	Diseño y construcción de un prototipo de un sistema de cerradura domiciliario automatizado.
Niveles de complejidad	<p>Nivel inicial: Construir, con impresión 3D y una Placa Arduino, una cerradura automatizada que se abra con un pulsador. El sistema también contará con 2 LED de color verde y rojo que indicarán si la cerradura está abierta o cerrada.</p> <p>Nivel intermedio: Agregar al sistema un teclado de membrana que permita establecer una clave para habilitar la apertura de la cerradura y un zumbador (<i>buzzer</i>) que indique si la clave ingresada por el usuario es correcta o incorrecta.</p> <p>Nivel avanzado: Monitorear la apertura de la cerradura desde un dispositivo móvil a través de Internet de las Cosas (IoT).</p>
Insumos	<ul style="list-style-type: none">● 20 cables dupont macho hembra● Filamento para impresora 3D (PLA)● 1 x Motor SG5010● 2 x Pulsadores● 1 x Teclado 4x4 de membrana● 1 x LED rojo 5mm difuso● 1 x LED verde 5mm difuso● 5 x resistencia 220Ω● 1x Buzzer● 1 x Arduino UNO R3● 1 x Protoboard● 1 x Cable USB tipo B● 1 x OBLOQ IOT MODULE● 1 x Fuente de 9V 1 A (plug centro positivo, 5.5x2.1mm)

Equipamiento	<ul style="list-style-type: none"> • Computadora • Soldador • Estaño • Alicates • Pinza de punta • Brusela
Otros requisitos	<ul style="list-style-type: none"> • Conexión a internet • Descargar el programa “mBlock3” • Descargar el programa “Arduino” http://www.mblock.cc/software-1/mblock/mblock3/

Presentación

Descripción ampliada del proyecto

Se propone el diseño, la construcción y la programación de un prototipo de una cerradura domiciliaria automática que no requiere de una llave para ser abierta. En una primera instancia, se automatizarán los procesos de apertura y cierre de la cerradura utilizando un pulsador y dos LED (uno rojo y uno verde) que indicarán el estado de la cerradura.

En el nivel intermedio, se propone incorporar un teclado de membrana que permita establecer una clave para abrir la cerradura y un zumbador (*buzzer*) que emita un sonido cada vez que alguien ingrese una clave, indicando si esta es correcta o incorrecta. En el nivel avanzado, se incorpora Internet de las Cosas (IoT) para monitorear a distancia la cerradura automática.

Al final de esta guía se puede encontrar un glosario donde se provee la información técnica necesaria para poder poner el proyecto en funcionamiento. El mismo cuenta con aclaraciones sobre los diversos elementos electrónicos involucrados así como también conceptos claves.

Objetivos

- Aproximarse al conocimiento y el manejo de distintos componentes electrónicos mediante la construcción de un prototipo de una cerradura automática.
- Construir circuitos eléctricos sencillos con el propósito de simular el funcionamiento de una cerradura automática.
- Conocer los componentes de la interfaz de Arduino.
- Analizar y desarrollar la programación de estructura secuencial de un programa de una cerradura automática que tenga LED indicadores (nivel inicial), un teclado para ingresar una clave y un zumbador (nivel intermedio).
- Aproximarse al funcionamiento de las impresoras y programas de impresión 3D.
- Utilizar una placa Protoboard para realizar pruebas rápidas del funcionamiento del circuito.

Soldar componentes electrónicos para el armado de la maqueta final.

Desarrollo

Nivel Inicial

Josefina visita a su abuelo todos los fines de semana. En el último tiempo, se dio cuenta de que al abuelo le lleva mucho tiempo encontrar la llave para abrir la puerta de su casa. Luego de pensarlo juntos, decidieron solucionar este problema construyendo una cerradura que se maneje con dos botones: uno para abrirla y otro para cerrarla. Además, van a agregarle otras funcionalidades. Por un lado, quieren que la cerradura haga un sonido al abrirse, para que alerte al abuelo y él pueda estar seguro de que se abrió. Por otro lado, van a agregarle dos luces (una verde y una roja), que indicarán si la cerradura quedó abierta o cerrada. Así será más fácil para el abuelo conocer rápidamente el estado de la puerta.

En esta primera instancia, se propone construir una cerradura automática que se abra cuando se presione un pulsador y se cierre cuando se presione otro. Las piezas del dispositivo se imprimirán en 3D y se armará un sistema con una placa Arduino y dos LED (uno rojo y uno verde) para indicar si la cerradura está abierta o cerrada.

Paso 1 - Realizar el montaje del servomotor en una Protoboard

El proyecto se inicia con el armado del circuito que permitirá controlar el movimiento de la cerradura que habilita la apertura de la puerta.

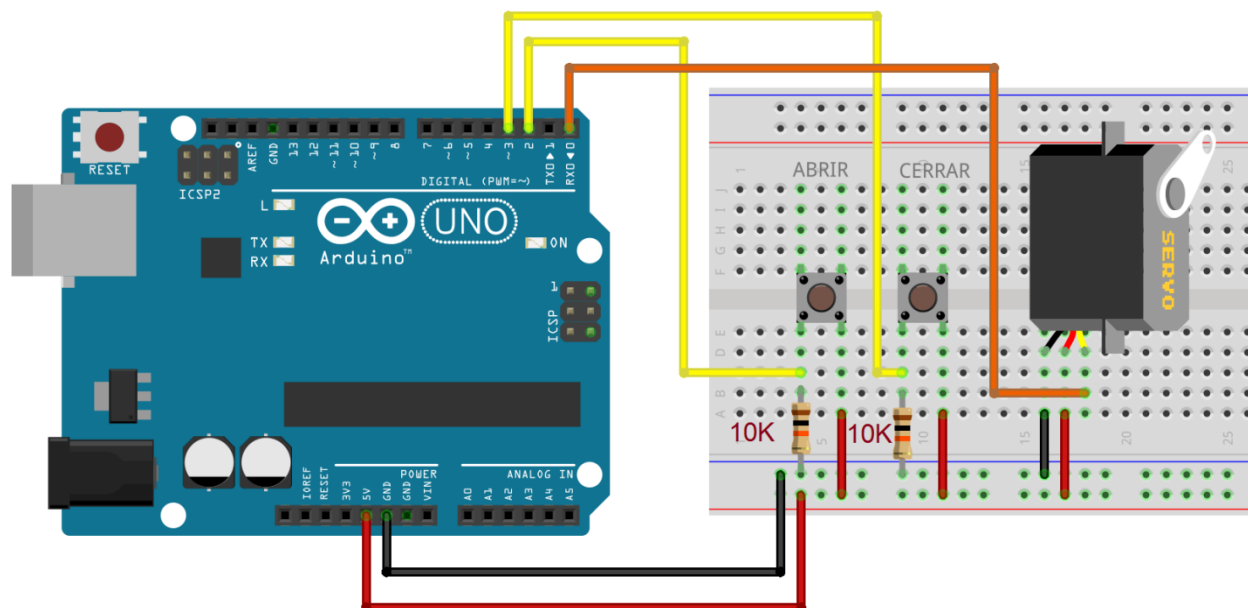
Conectamos un servomotor y los pulsadores a la placa Arduino. Armaremos un programa para que, cuando se presione un pulsador, el servomotor gire 180° y mueva el mecanismo que destraba la cerradura. Al presionar el otro pulsador, el servomotor girará en el sentido contrario, volviendo a trabar la cerradura.

En primer lugar, se explicará cómo armar el circuito y programar el sistema de apertura para la cerradura. Más adelante agregaremos las piezas mecánicas impresas en 3D para el armado de la maqueta.



Un servo es un dispositivo que se compone de un motor y un sistema de control que le permite ubicarse en una posición específica. Los servos más comunes pueden moverse en un rango de 0° a 180°, sin poder girar de forma continua. Se suelen utilizar en aplicaciones tipo barreras o brazos mecánicos.

Conectamos el servomotor a la Protoboard como indica la figura:



Esquema 1

Paso 2 - Programar el movimiento del servomotor

La programación la realizaremos con mBlock3, entorno de programación basado en Scratch2 que permite programar proyectos de Arduino utilizando bloques. Pueden descargarlo siguiendo este enlace: <http://www.mblock.cc/software-1/mblock/mblock3/>

Cuando abrimos mBlock3, veremos una pantalla como la siguiente:

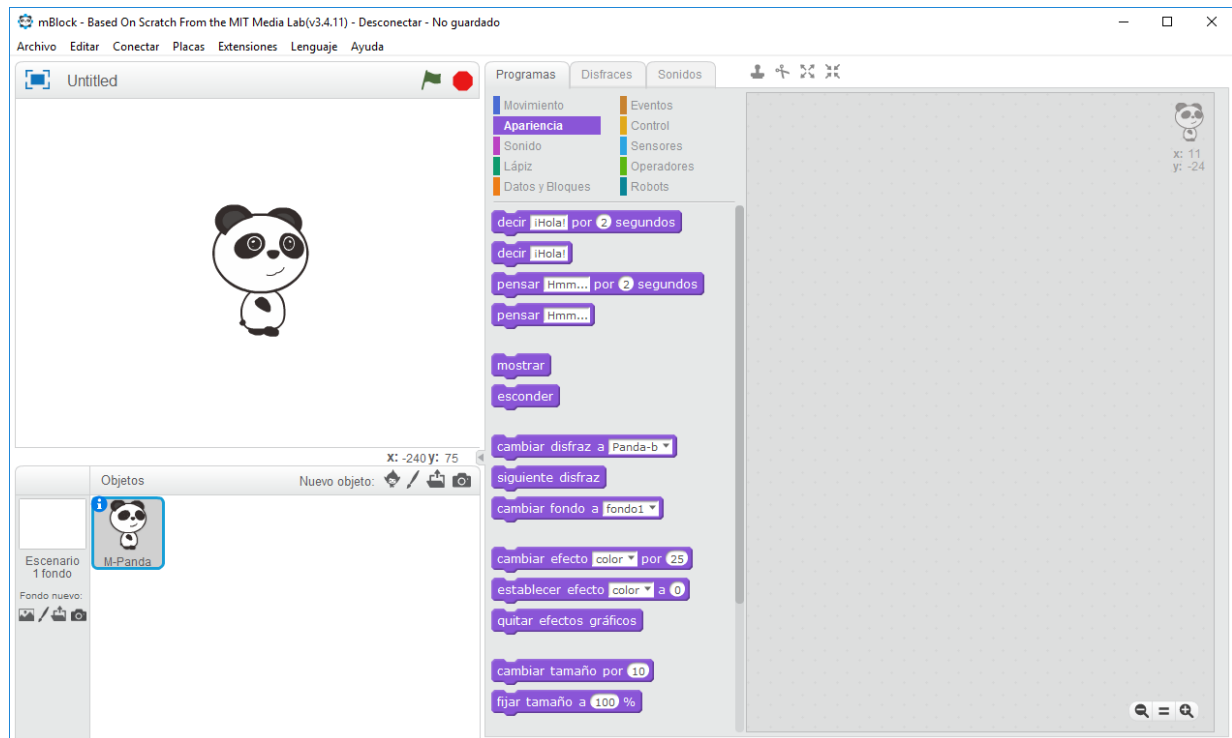


Fig.1

Para programar un proyecto de Arduino con mBlock3 debemos seleccionar el “Modo Arduino” desde el menú.

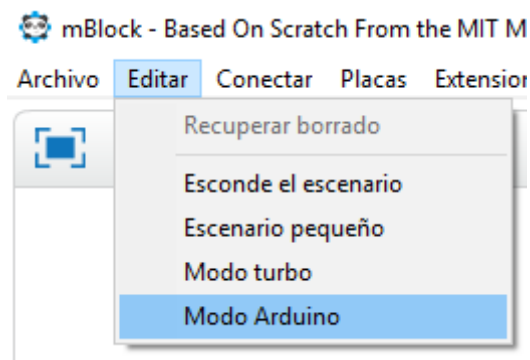


Fig.2

Al seleccionar este modo, el programa cambiará de aspecto. Se verá un área en el centro que es la que utilizaremos para programar con bloques. A la derecha se verá un campo donde aparecerá el código escrito que le corresponde a los bloques que están en el centro. Este código se irá escribiendo automáticamente a medida que se vaya armando el programa con los bloques.

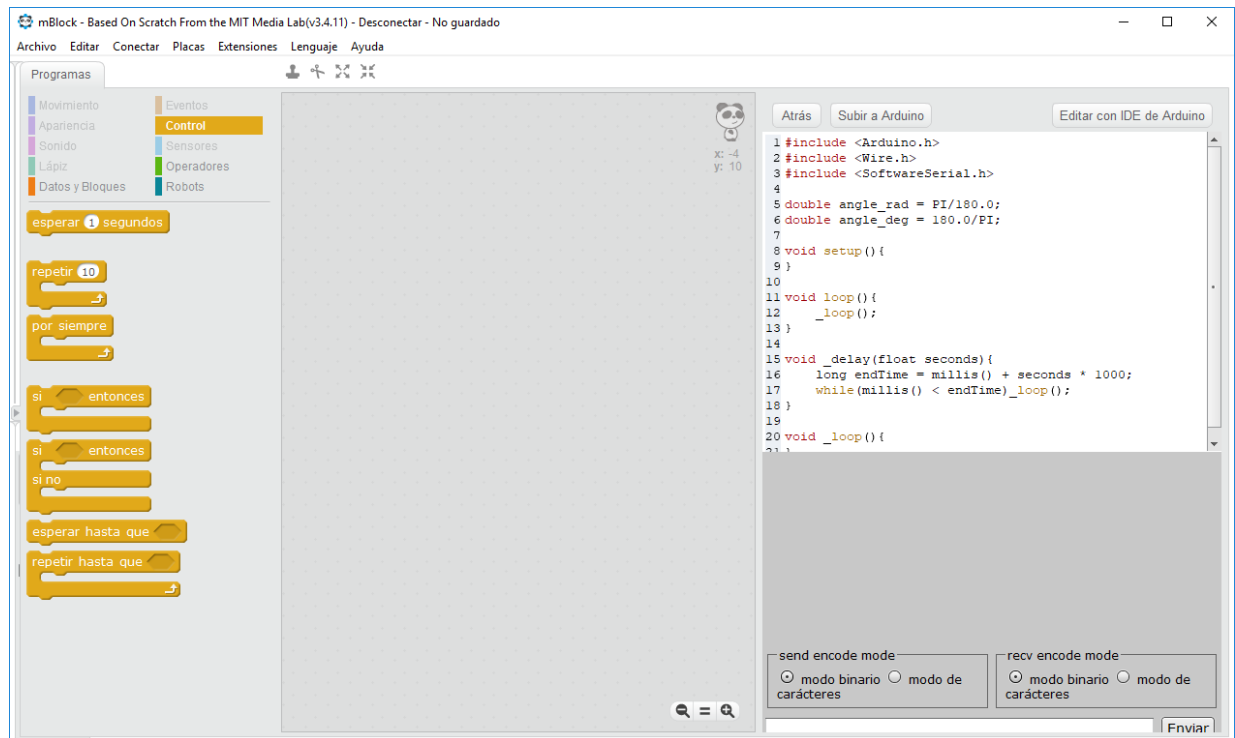


Fig.3

Los bloques están agrupados por categorías. En este caso, se usarán bloques de las categorías “**Robots**”, “**Control**”, “**Operadores**”, y “**Datos y Bloques**”. Cuando seleccionamos una de estas categorías, se pueden visualizar todos los bloques que pertenecen a ese grupo.



Fig.4

Primero, se programará el movimiento del servomotor de manera que responda a los siguientes comandos:

- Si se presiona el pulsador de apertura (*pin 2*) el servomotor se moverá hasta completar un ángulo de 180° .
- Si se presiona el pulsador de cierre (*pin 3*) el servomotor se moverá hasta volver al ángulo original de 0° .

Al estado de reposo de los pulsadores (es decir, cuando no están siendo accionados) le asignaremos el valor "0" y al estado activo (es decir, cuando están siendo accionados) le asignaremos el valor "1".

El programa nos quedaría entonces similar al siguiente modelo:



Fig.5

Veremos que a continuación se muestra el código escrito generado por mBlock, que corresponde a este programa:

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

#include <Servo.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
Servo servo_0;

void setup(){
  pinMode(2,INPUT);
  servo_0.attach(0); // init pin
  pinMode(3,INPUT);
}

void loop(){
  if(((digitalRead(2))==(1))){
```

```

    servo_0.write(180); // write to servo
}
if(((digitalRead(3))==(1 ))){
    servo_0.write(0); // write to servo
}
_loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}

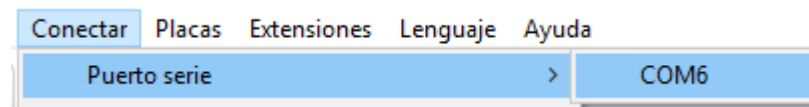
void _loop(){
}

```

Paso 3 - Subir el código a la placa Arduino

Para subir el código de nuestro programa a la placa Arduino, necesitamos:

1. Conectar la placa Arduino a la entrada USB.
2. Chequear que, en el menú "Placas", esté seleccionado "Arduino Uno".
3. Seleccionar el puerto serie al que está conectada la placa.

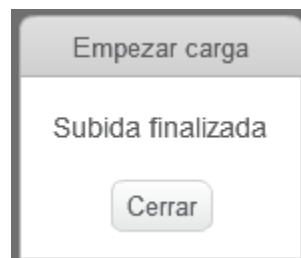


4. Clickear el botón

Subir a Arduino

Al terminar de subir nuestro

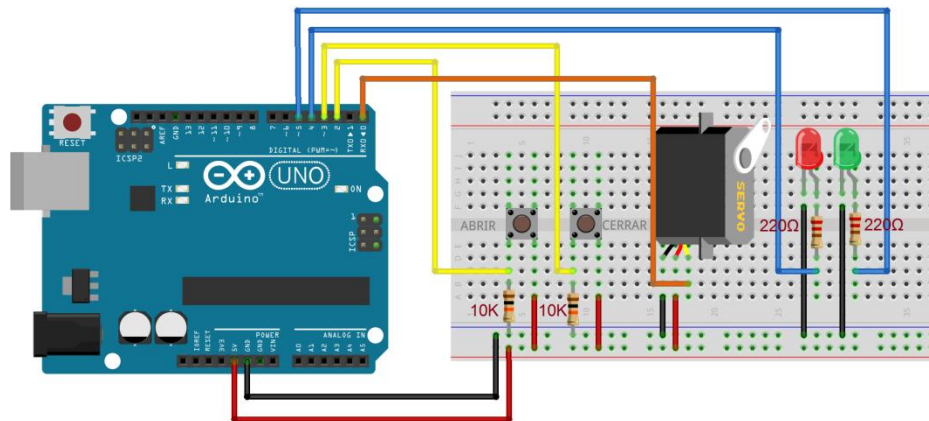
código, veremos este mensaje:



Paso 4 - Conectar los LED indicadores

En este momento, agregaremos a nuestro sistema 2 LED (uno rojo y uno verde) que indicarán si la cerradura está abierta (verde) o cerrada (rojo). Debemos conectarlos como indica la figura:

En un LED, la pata larga siempre es la “pata positiva” y es donde se conecta a nuestro *pin*.



Esquema 2

Paso 5 - Programar los LED indicadores

Luego de conectar los LED a la Protoboard, modificaremos el programa que realizamos previamente para que ahora indique si la cerradura está abierta o cerrada determinando el encendido del LED correspondiente en cada caso. Establecemos que si la cerradura está cerrada se encienda el LED rojo y si está abierta se encienda el LED verde.

El programa debe ser similar al siguiente:

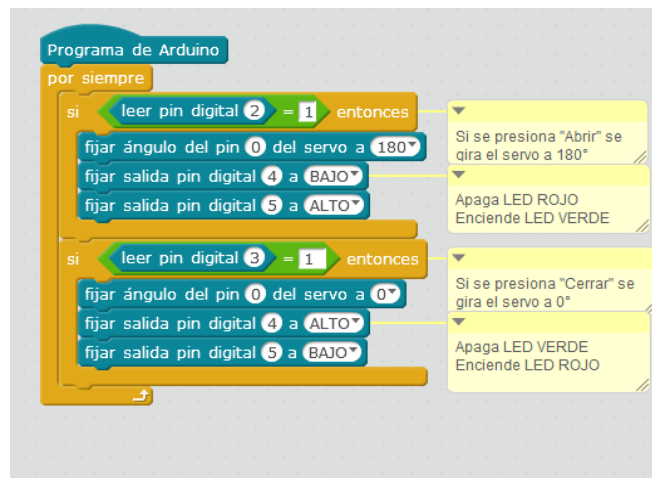


Fig.6

A continuación podemos observar el código escrito que corresponde a este programa.

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

#include <Servo.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
Servo servo_0;

void setup(){
    pinMode(2,INPUT);
    servo_0.attach(0); // Inicio el servomotor
    pinMode(4,OUTPUT);
    pinMode(5,OUTPUT);
    pinMode(3,INPUT);
}

void loop(){
    if(((digitalRead(2))==1))){
        servo_0.write(180); // Ubico al servomotor en la posición 180°
        digitalWrite(4,0);
        digitalWrite(5,1);
    }
    if(((digitalRead(3))==1 ))){
        servo_0.write(0); // Ubico al servomotor en la posición 0°
        digitalWrite(4,1);
        digitalWrite(5,0);
    }
    _loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}

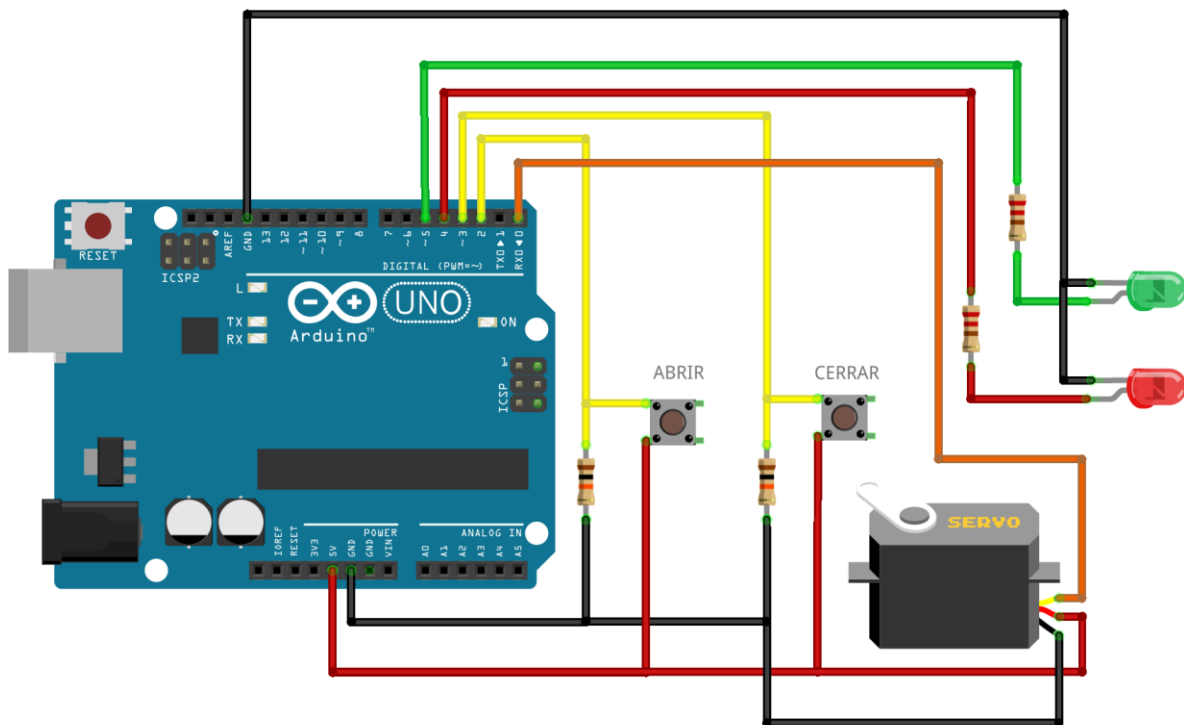
void _loop(){
}
```

Paso 6 - Comenzar a armar la maqueta

En este momento, debemos decidir si queremos avanzar al nivel de complejidad intermedio o no. Si decidimos cerrar el proyecto en este nivel, continuamos en este paso para armar la maqueta de nuestro sistema. Si, por el contrario, queremos avanzar con el proyecto e incluir un teclado de membrana, pasamos al próximo nivel.

Si decidimos avanzar con la maqueta, procederemos a soldar los componentes, imprimir las piezas 3D y ensamblar todo el conjunto.

Una vez que las conexiones han sido probadas y el sistema funciona correctamente, podemos pasar a armar los circuitos prescindiendo de la Protoboard. Para hacerlo, debemos soldar todos los cables a los componentes como indica el siguiente esquema:



Esquema 3

Paso 7 - Imprimir las piezas

Ahora que tenemos las conexiones soldadas, vamos a imprimir las piezas del sistema mecánico y ensamblar todo el conjunto.

Para armar la maqueta del semáforo, se deben imprimir las piezas en la impresora 3D. El modelo 3D del semáforo se puede descargar de forma libre y gratuita en el siguiente enlace: <https://enfoco.net.ar/sd>

Una vez descargado el modelo, lo imprimimos con la impresora 3D. Cuando estén listas todas las piezas, las ensamblamos para construir el sistema mecánico de la cerradura.

En caso de querer realizar una modificación en el modelo, independientemente del programa de modelado que utilicemos, debemos exportar nuestras piezas en formato .stl.

El .stl es un formato de archivo informático de diseño asistido por computadora (CAD) que define la geometría de objetos sólidos 3D. Es el formato más popular a la hora de intercambiar digitalmente modelos de objetos para ser impresos en 3D.

Las piezas son las siguientes:

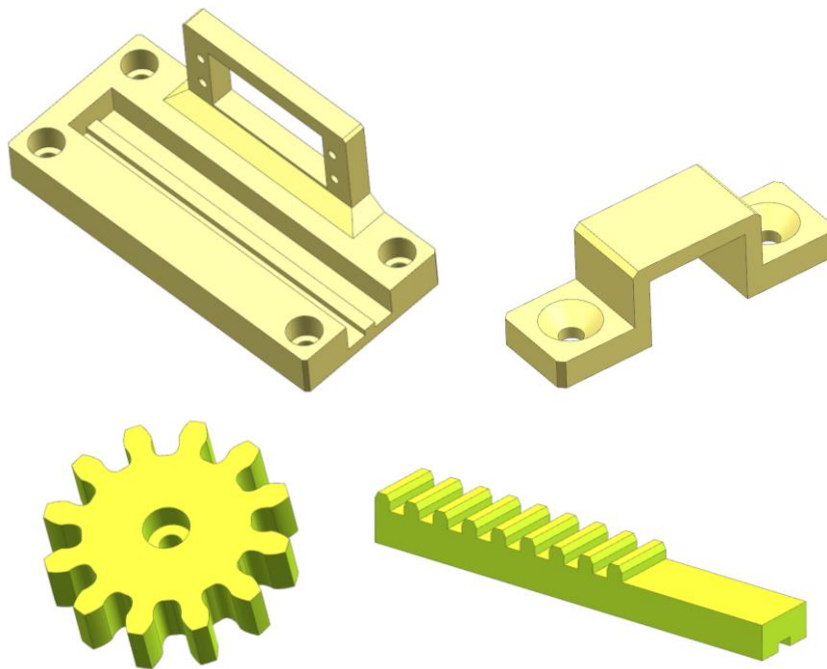


Fig. 7

Paso 8 - Ensamblar el sistema mecánico con el servomotor

Una vez que se imprimieron todas las piezas, estas se deben ensamblar entre sí con el servomotor como se ve en la imagen que se muestra a continuación. En este esquema se presenta solo la conexión al servomotor, pero es necesario conectar también la placa Arduino, los pulsadores y los LED.

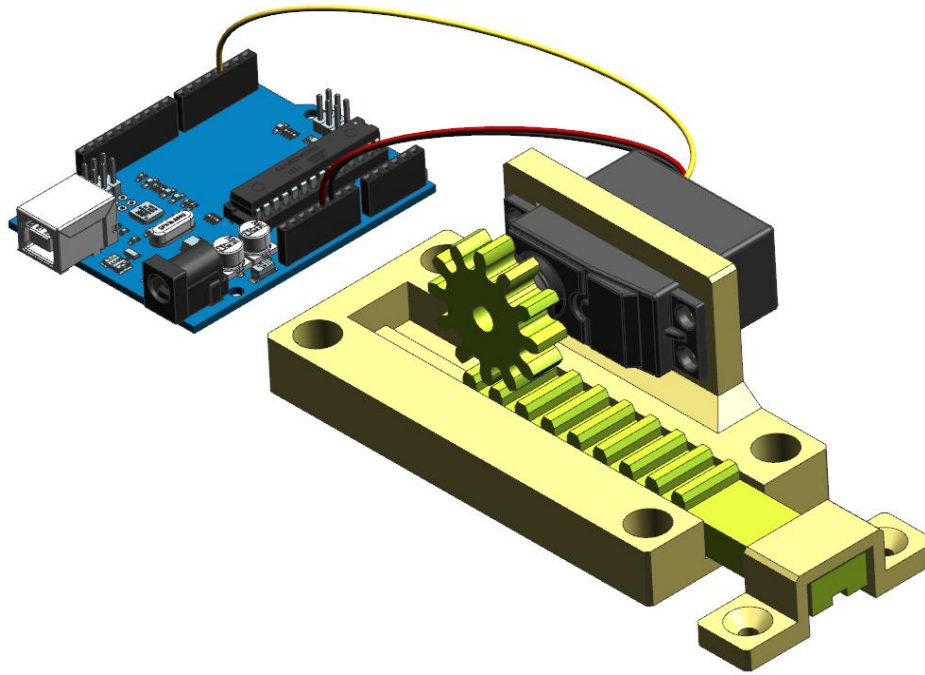


Fig. 8

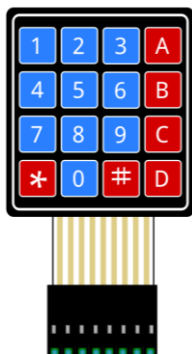
Una vez armado el sistema, se puede fijar a la puerta (o apertura similar) que se desee y ubicar allí los LED y pulsadores.

Nivel Intermedio

Por un tema de seguridad, ahora Josefina quiere que la cerradura se abra con una clave y no simplemente oprimiendo un botón. Por lo tanto, va a incluir en el sistema un teclado que le permita determinar la clave e ingresarla cada vez. Además, va a incluir también un dispositivo que sonará cada vez que se ingrese una clave en el teclado, indicando si esta es correcta o incorrecta.

El abuelo está de acuerdo con las modificaciones, pero le pidió que conserve el sistema de LED que señalizan si la puerta está abierta o cerrada, dado que le resultó muy útil.

Se propone agregar al sistema de la cerradura dos nuevos dispositivos: un teclado de membrana, que permita establecer una clave para habilitar su apertura; y un zumbador (*buzzer*), que emita un sonido cada vez que se ingresa un código correcto y otro cuando se ingresa uno incorrecto.



El teclado de membrana está compuesto por dos láminas flexibles con circuitos impresos sobre una tinta conductora. Al presionar una tecla, las membranas hacen contacto cerrando el circuito.

Paso 1 - Subir el código a través de Arduino IDE.

La programación por bloques tiene sus ventajas desde un punto de vista didáctico pero cuando el programa crece en complejidad puede resultar poco práctico. A menudo podemos encontrarnos con el hecho de que ciertas operaciones no pueden resolverse utilizando bloques o que hacerlo con este método resulta más engorroso y difícil de interpretar que si se utilizara el código escrito.

Hasta ahora hemos visto cómo al realizar nuestra programación en bloques se generaba simultáneamente un código escrito en el área lateral derecha. Para esta sección de la actividad se propone trabajar directamente sobre el código, para ello vamos a recurrir a el entorno nativo de Arduino que llamamos Arduino IDE (entorno de desarrollo integrado).

Para ello descarga el Arduino IDE desde el siguiente enlace y luego procede con la instalación del mismo: www.enfoco.net.ar/sd

Veremos que se nos presenta la siguiente interfaz:

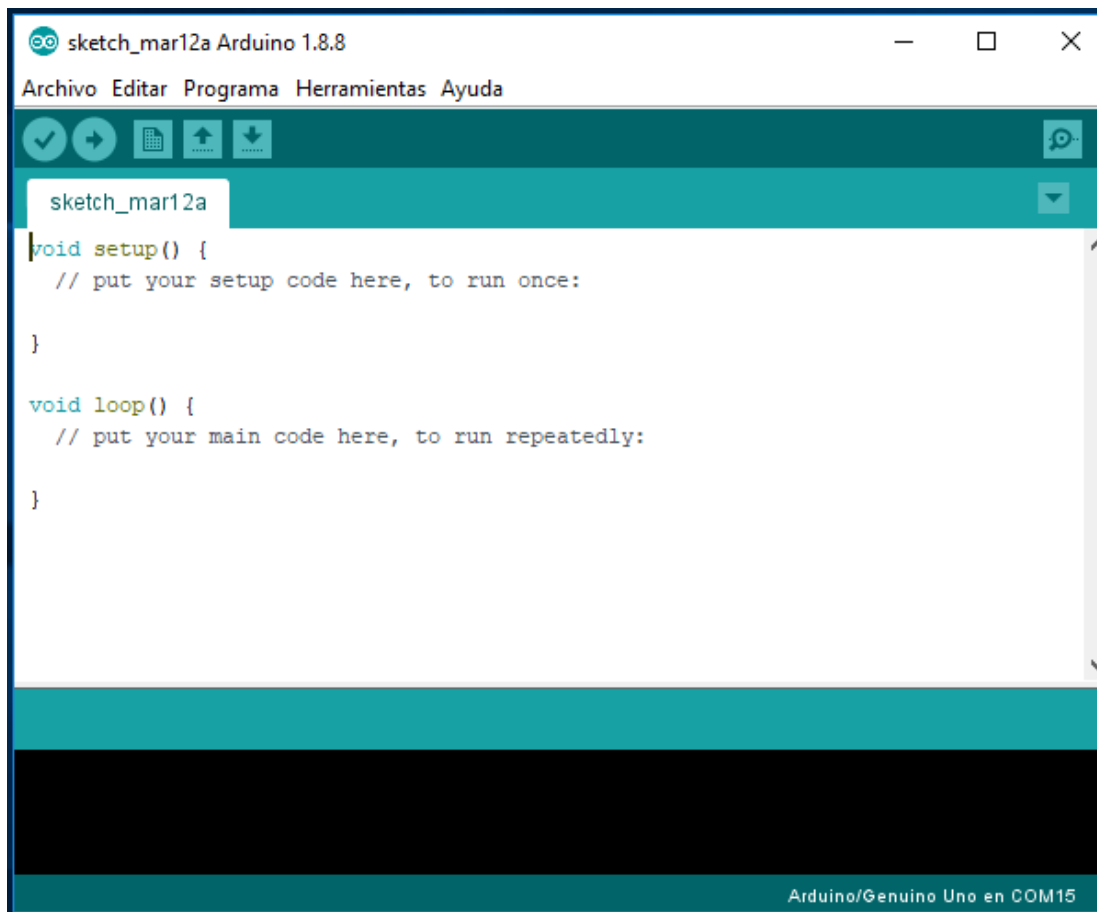


Fig. 9

A continuación, se presenta una estructura mínima de un *sketch* (un programa) de Arduino:

```
void setup() {  
  // Código de inicialización. Se ejecuta una sola vez.  
}  
  
void loop() {  
  // Código principal. Se ejecuta repetidamente.  
}
```

En líneas generales, un programa de Arduino es:

1. Un bloque de código que se ejecuta por única vez al inicializarse el dispositivo. Este bloque de código está contenido dentro de la función “setup” (se coloca dentro de `void setup() { y }`).

2. Un bloque de código que se ejecuta repetidamente luego de la función “setup”. Este bloque de código está contenido dentro de la función “loop” (se coloca dentro de `void loop() { y }`).

Después de `//` se incluyen comentarios para el lector que no tienen ningún efecto en el programa. Estos comentarios sirven para clarificar el código y que sea más fácil de interpretar para otras personas.

Los pasos para subir el código a través del Arduino IDE son similares a los que hemos visto para mBlock3:

1. Conectar la placa a la entrada USB.
2. Chequear que estén seleccionados la placa “Arduino/Genuino Uno” y el puerto serie al que está conectada la placa.

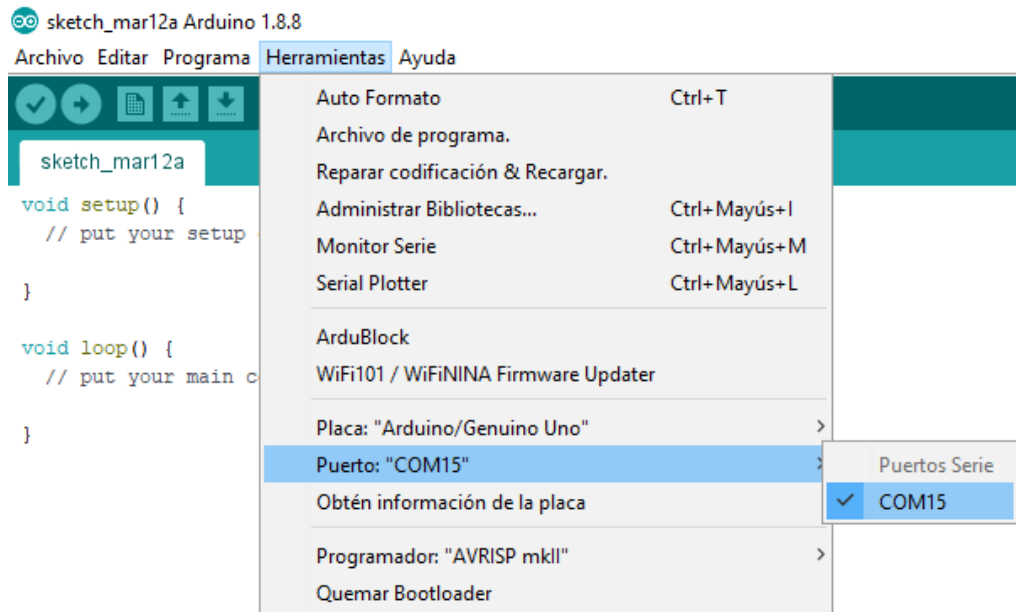


Fig. 10

3. Clickear el botón de “Subir”  para comprobar los paso anteriores.

Sabremos que nuestro código subió correctamente si en la barra de estado se escribe “Subido”.

Paso 2 - Incluir librería “Keypad”

Es necesario instalar la librería “Keypad” en nuestro programa Arduino. Para hacerlo seleccionamos “Programa” en el menú. Luego, “Incluir librería”. Y, por último, “Gestionar librería”.

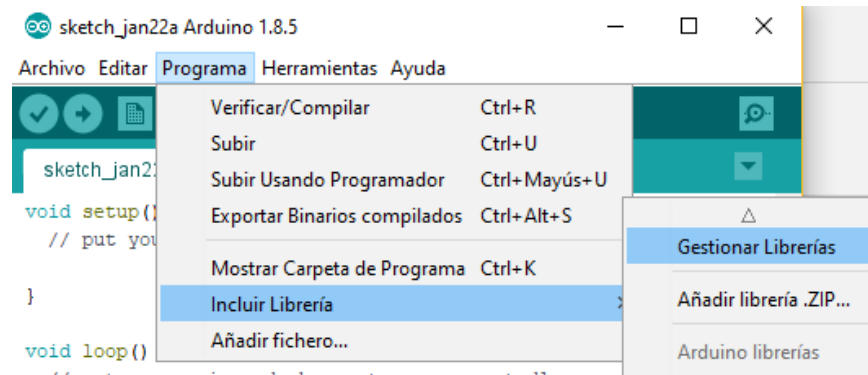


Fig.11

Buscamos la librería “Keypad” y seleccionamos “Instalar”

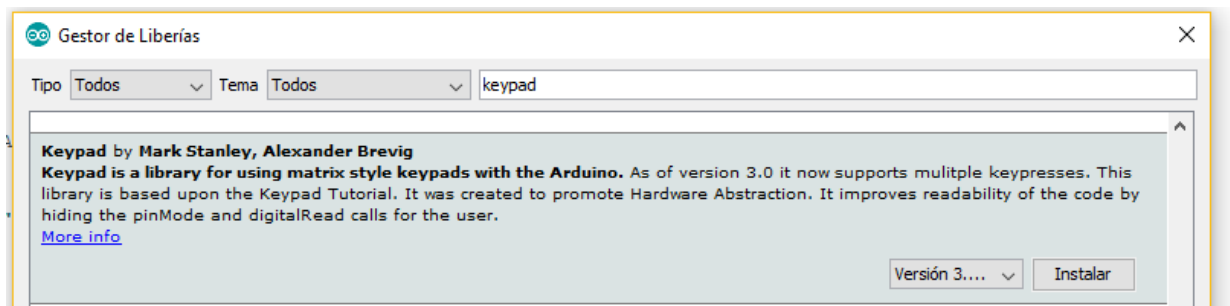
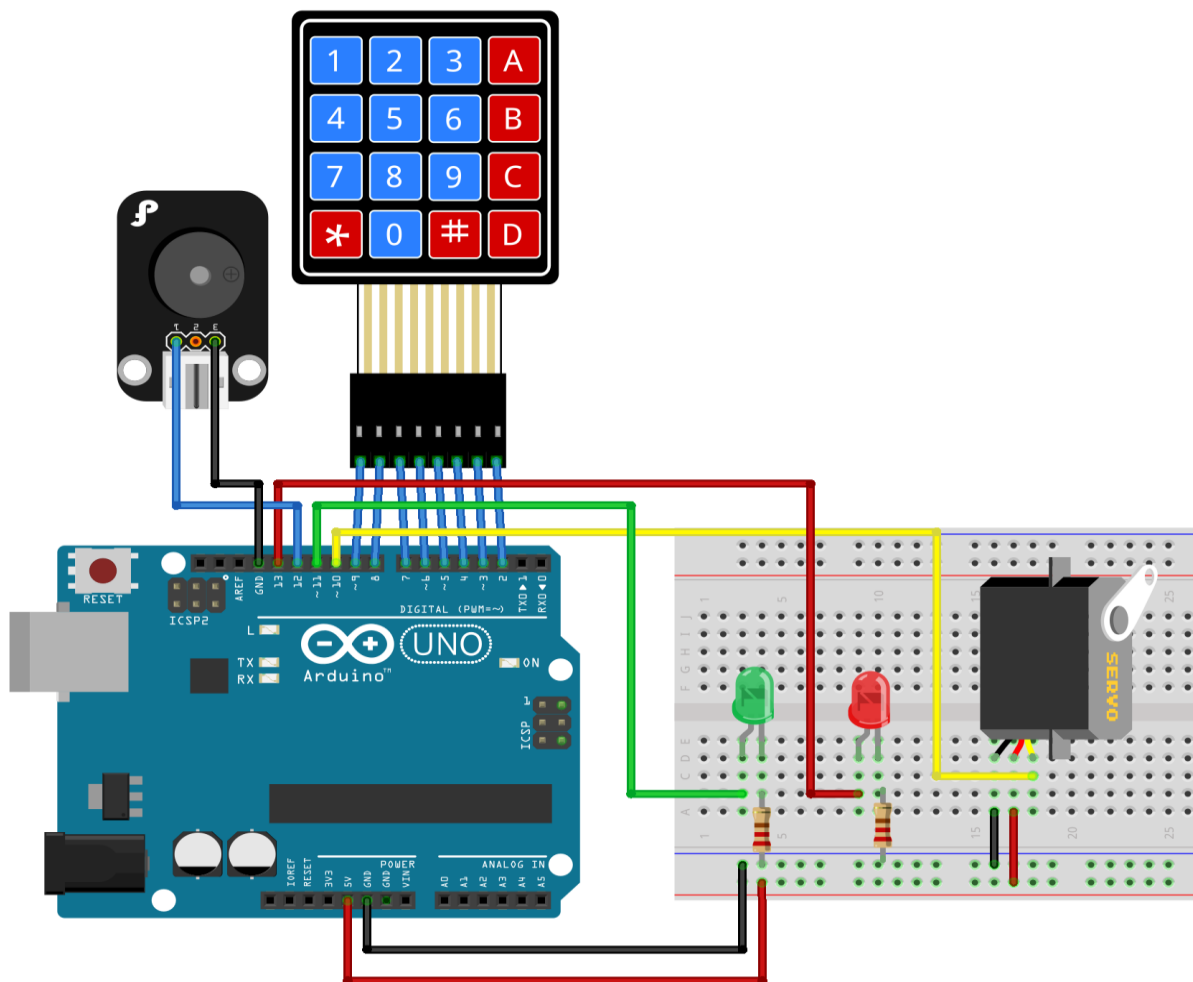


Fig.12

Conectamos el teclado de membrana y el zumbador en el Arduino y el servo motor en la Protoboard, como indica el siguiente esquema:

Un **Zumbador** (*buzzer*) es un dispositivo que genera sonido con una frecuencia determinada. Al recibir alimentación eléctrica comienza a sonar, controlando la misma se pueden generar sonidos continuos o pulsos. Sirve como mecanismo de señalización o aviso.



Esquema 4

Paso 3 - Programando el teclado de membrana

Lo primero que vamos a establecer es la cantidad de dígitos que debe tener nuestra contraseña. A modo de ejemplo, en nuestro caso tendrá 4 dígitos y la clave será “1 5 9 D”.

Una matriz es una lista de datos. Es posible tener una matriz de cualquier tipo de datos. Cada pieza de datos en una matriz se identifica mediante un número de índice que representa su posición en la matriz. El primer elemento de la matriz es [0], el segundo elemento es [1] y así sucesivamente.

Existen matrices de diversas formas, las de una sola dimensión se las puede representar con una lista de datos, las de 2 dimensiones equivalen a una tabla con filas y columnas, las de 3 dimensiones en adelante ya no son posibles representarla de manera gráfica directa.

Cada matriz tiene una longitud variable, que es un valor entero para el número total de elementos en la matriz. Hay que tener en cuenta que, dado que la numeración del índice comienza en cero (no 1), el último valor en una matriz con una longitud de 5 debe ser

referenciado como matriz [4] (es decir, la longitud menos 1), no matriz [5] , lo que desencadenaría un error.

0	1	2	3	
1	5	9	D	Matriz

La contraseña la guardaremos en una matriz que establecerá cuáles son los dígitos que permiten la apertura de la cerradura.

```
int Clave[4]={'1','5','9','D'};
```

Cuando el sistema detecte que se presionaron 4 dígitos en el teclado,

```
int Clave_Pres[4]={'0','0','0','0'};
```

comparará el código ingresado con el código correcto guardado en la matriz. Si coinciden, permitirá el movimiento del motor -que abrirá la cerradura-, activará el zumbador y encenderá la luz verde.

```
if(Clave[0]==Clave_Pres[0] && Clave[1]==Clave_Pres[1] && Clave[2]==Clave_Pres[2]  
&& Clave[3]==Clave_Pres[3]){
```

Si no coincide, el programa volverá a empezar. Es decir, volverá a estar disponible para que se presionen 4 caracteres nuevos.

```
}else{//Código ingresado incorrecto, emite doble pitido.  
    digitalWrite(12, HIGH); //Enciende el buzzer.  
    delay(200);  
    digitalWrite(12, LOW); //Apaga el buzzer.  
    delay(300);  
    digitalWrite(12, HIGH); //Enciende el buzzer.  
    delay(200);  
    digitalWrite(12, LOW); //Apaga el buzzer.
```

El código completo quedaría como se ve a continuación:

```
#include <Keypad.h>  
#include <Servo.h>  
  
Servo Cerradura; // Se crea la variable Cerradura (servomotor).
```

```

const byte Filas = 4; //4 Filas.
const byte Columnas = 4; //4 Columnas.
//Definición de la matriz de teclas del teclado.
char Teclas[Filas][Columnas] = {
    {'1','2','3','A'},
    {'4','5','6','B'},
    {'7','8','9','C'},
    {'*','0','#','D'}
};
byte Pins_Filas[Filas] = {9, 8, 7, 6}; //Conectar los pines de las filas del
teclado.
byte Pins_Columnas[Columnas] = {5, 4, 3, 2}; //Conectar los pines de las
columnas del teclado.

//Se inicia el teclado.
Keypad Teclado = Keypad( makeKeymap(Teclas), Pins_Filas, Pins_Columnas, Filas
Columnas);

void setup(){
    pinMode(11,OUTPUT); //Configura el pin 11 como salida LED VERDE.
    pinMode(12,OUTPUT); //Configura el pin 12 como salida BUZZER.
    pinMode(13,OUTPUT); //Configura el pin 13 como salida LED ROJO.
    Cerradura.attach(10); //Configura el servomotor de la cerradura.
    Cerradura.write(0); //Inicia la apertura de la cerradura.
    digitalWrite(11, HIGH); //Enciende LED VERDE.
    Serial.begin(9600);
}

int Clave[4]={'1','5','9','D'}; // Clave para acceder a la puerta.
int Clave_Pres[4]={'0','0','0','0'}; //Clave presionada por el usuario.
int indice=0;
//estado_puerta: Si el valor es "1", la puerta está abierta.
// Si el valor es "0", la puerta está cerrada.
// Inicialmente está abierta.
int estado_puerta=1;
void loop(){
    char Tecla_Pres = Teclado.getKey(); //Lee la tecla presionada.

    if (Tecla_Pres){ //Si es una tecla válida, la guarda.
        Clave_Pres[indice]=Tecla_Pres;
        indice++; //Incrementa la cantidad de teclas presionadas.
        if(indice==4){ //Si las teclas presionadas son 4, compara las claves.
            indice=0;

```

```

    if(Clave[0]==Clave_Pres[0] && Clave[1]==Clave_Pres[1] &&
Clave[2]==Clave_Pres[2] && Clave[3]==Clave_Pres[3]){
        digitalWrite(12, HIGH); //Enciende el buzzer.
        if(estado_puerta==0){//Si la cerradura está cerrada, la abre.
            Cerradura.write(0);
            estado_puerta=1;
            digitalWrite(11, HIGH); //Enciende LED VERDE.
            digitalWrite(13, LOW); //Apaga LED ROJO.
        }else{//Si la puerta está abierta, la cierra
            Cerradura.write(90);
            estado_puerta=0;
            digitalWrite(11, LOW); //Apaga LED VERDE.
            digitalWrite(13, HIGH); //Enciende LED ROJO.
        }
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer.
    }else{//Si el código ingresado es incorrecto, emite doble pitido.

        digitalWrite(12, HIGH); //Enciende el buzzer.
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer.
        delay(300);
        digitalWrite(12, HIGH); //Enciende el buzzer.
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer.

    }
}
}
}
}

```


Nivel Avanzado

A Josefina se le ocurrió que sería bueno aumentar aún más el nivel de seguridad de la cerradura. Quiere que su abuelo pueda saber cuándo se abre o se cierra la puerta aunque no se encuentre en su casa, para que se entere si en algún momento hay alguien entrando allí en su ausencia. Para conseguirlo, decidió agregarle a la cerradura un sistema de Internet de las Cosas (IoT), que le permita monitorear esta información desde un dispositivo móvil.

A través de Internet de las Cosas (IoT) se podrá monitorear desde un dispositivo móvil si la cerradura está abierta o cerrada.

Paso 1 - Introducción a Internet de las Cosas (IoT)

Internet de las Cosas (en inglés *Internet of Things*, abreviado IoT) es un concepto que refiere a la interconexión digital de objetos cotidianos con internet. Esta interconexión puede tener diversas funciones. Por ejemplo, puede utilizarse para monitorear la temperatura de un ambiente, enviando los datos obtenidos por un sensor a una central donde se recopile la información. De esta manera podría visualizarse en un dispositivo móvil la temperatura de un laboratorio, de un invernadero o de una sala de un hospital. Para poder incorporar IoT a nuestro proyecto es necesario:

1. Un dispositivo capaz de conectarse a internet.
2. Un servidor que reciba y aloje los datos.

Existen diversas formas de lograr el cometido de registrar y almacenar los datos del sistema de tanques construido. En este caso, se detallará cómo hacerlo con un módulo OBLOQ de DFRobot, y con los servidores de Adafruit.

El módulo UART OBLOQ es un dispositivo WiFi a serie pensado para desarrolladores no profesionales. Permite enviar y recibir datos mediante los protocolos HTTP y MQTT.

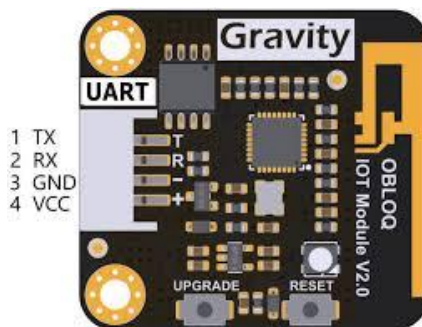


Fig. 13

Paso 2 - Crear un Panel de Control

En primer lugar, se explicará cómo crear un Panel de Control en Adafruit. Luego, se verá cómo vincular los controles del Panel con los datos que se intercambian con el dispositivo. Primero debemos crear una cuenta de usuario en io.adafruit.com.

Una vez que ingresamos con nuestro usuario, creamos un nuevo panel haciendo click en el botón "Actions" y seleccionamos "Create a New Dashboard".

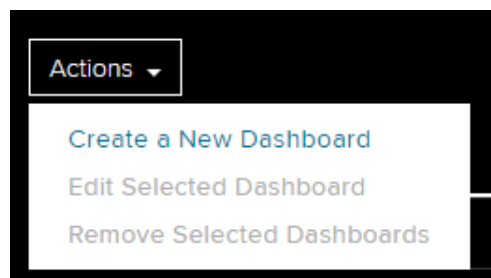
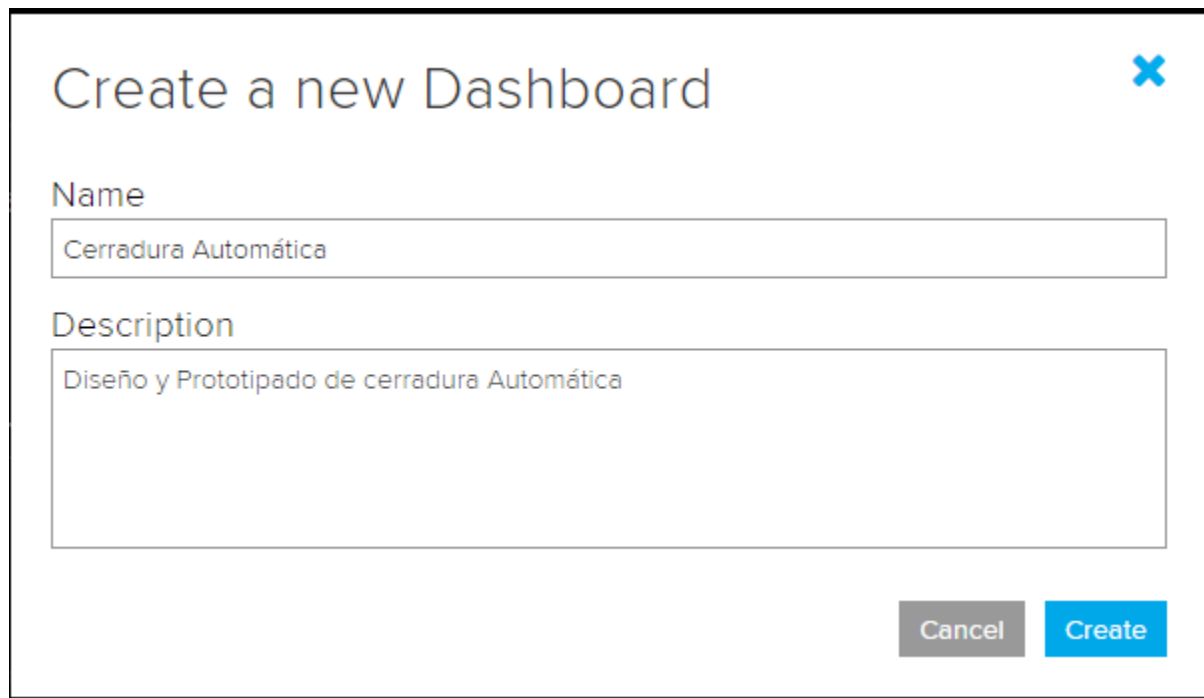


Fig. 14

Luego designamos un nombre y una descripción. Y al finalizar presionamos el botón "Create".



The image shows a dialog box titled "Create a new Dashboard" with a blue 'X' icon in the top right corner. It contains two input fields: "Name" with the text "Cerradura Automática" and "Description" with the text "Diseño y Prototipado de cerradura Automática". At the bottom right, there are two buttons: "Cancel" (grey) and "Create" (blue).

Fig. 15

Seguidamente hacemos click en el nuevo panel creado y veremos una pantalla vacía.


Podemos comenzar a agregar bloques haciendo click en .



Fig. 16

Veremos una serie de controles posibles como en la siguiente imagen:

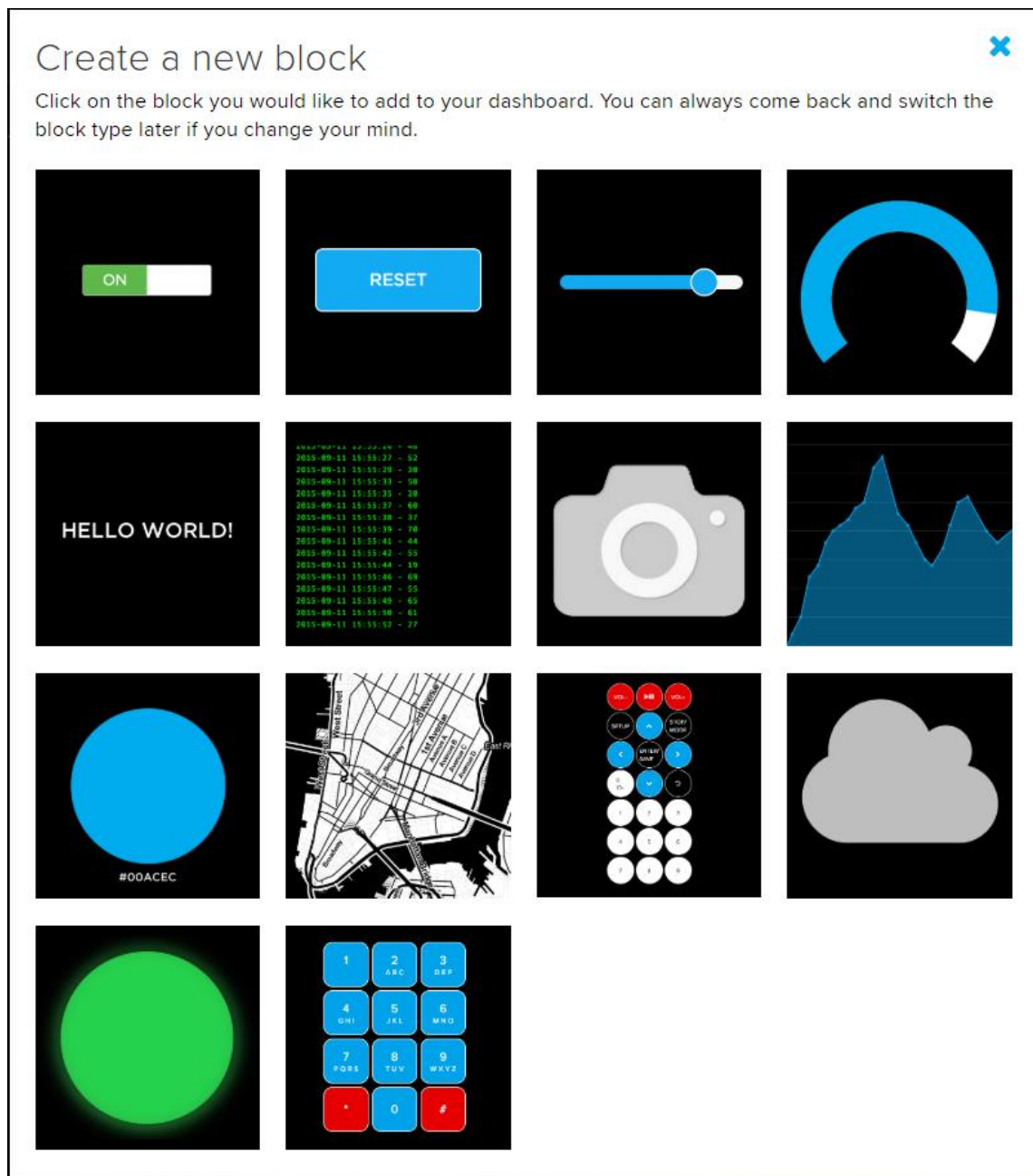


Fig. 17

Para nuestro proyecto, podríamos ubicar un *Indicator* (indicador) con diferentes colores que muestre el estado de la cerradura.



Fig. 18

Cuando agregamos un control al panel debemos asociarlo a un “feed”.

Un **feed** es una fuente de datos en la que uno puede publicar así como también se puede suscribir para recibir los datos de cierto feed.

Llamamos al primer *feed* “cerradura”. En él publicaremos, desde nuestro dispositivo, la información sobre el estado de la cerradura.

Choose feed ✕

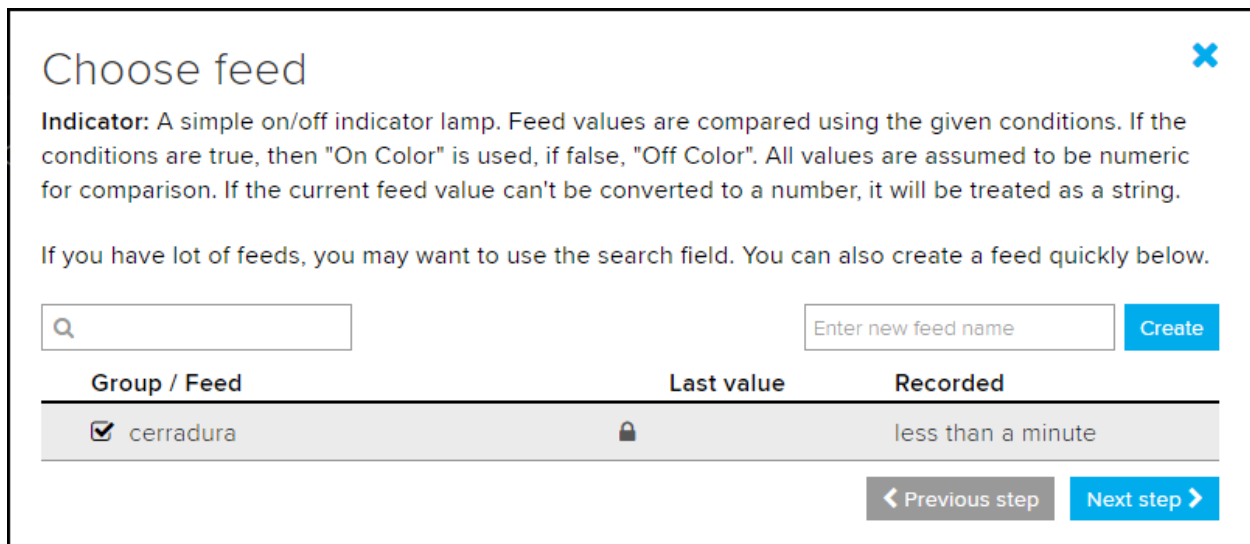
Indicator: A simple on/off indicator lamp. Feed values are compared using the given conditions. If the conditions are true, then "On Color" is used, if false, "Off Color". All values are assumed to be numeric for comparison. If the current feed value can't be converted to a number, it will be treated as a string.

If you have lot of feeds, you may want to use the search field. You can also create a feed quickly below.

Group / Feed	Last value	Recorded
<input type="checkbox"/> Welcome Feed		2 days

Fig. 19

Luego de crearlo, hacemos click en “*Next step*” (paso siguiente) para configurar nuestro control.



Choose feed ✕

Indicator: A simple on/off indicator lamp. Feed values are compared using the given conditions. If the conditions are true, then "On Color" is used, if false, "Off Color". All values are assumed to be numeric for comparison. If the current feed value can't be converted to a number, it will be treated as a string.

If you have lot of feeds, you may want to use the search field. You can also create a feed quickly below.

Create

Group / Feed	Last value	Recorded
<input checked="" type="checkbox"/> cerradura		less than a minute

< Previous step Next step >

Fig.20

Este *feed* tendrá el estado de la cerradura expresado con la siguiente convención:

- Cuando el valor sea “0” estará en rojo.
- Cuando el valor sea “1” estará en verde.

Podemos incluir un título, un color que indique el estado activado (cuando la cerradura está abierta) y otro color que indique el estado desactivado (cuando la cerradura está cerrada). Este indicador estará en estado activado cuando se cumpla la condición determinada. En este caso, haremos que el indicador esté en color verde (activado) cuando el *feed* “cerradura” sea igual a 1. Cuando sea distinto de 1 estará en color rojo (desactivado).

Para configurarlo, completamos los campos como se ve en la imagen a continuación.

Block settings

In this final step, you can give your block a title and see a preview of how it will look. Customize the look and feel of your block with the remaining settings. When you are ready, click the "Create Block" button to send it to your dashboard.

Block Title (optional)

Cerradura

On Color

#00ff00

Off Color

#ff0000

Conditions

=

1

Add Condition

Block Preview

Cerradura

Indicator

A simple on/off indicator lamp. Feed values are compared using the given conditions. If the conditions are true, then "On Color" is used, if false, "Off Color". All values are assumed to be numeric for comparison. If the current feed value can't be converted to a number, it will be treated as a string.

Test Value

1

Previous step

Create block

Fig. 21

Al finalizar la configuración, hacemos click en "Create block" (crear bloque) para completar la operación.

Podemos modificar el tamaño y la ubicación de los bloques haciendo click en la "rueda de configuración".

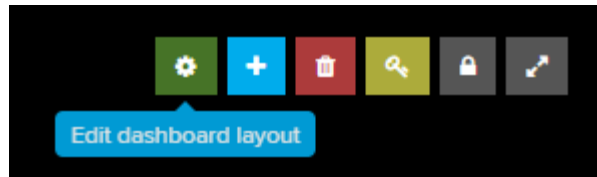


Fig.22

Al finalizar la configuración de los colores para el estado de la cerradura, deberíamos visualizar algo similar a lo siguiente:

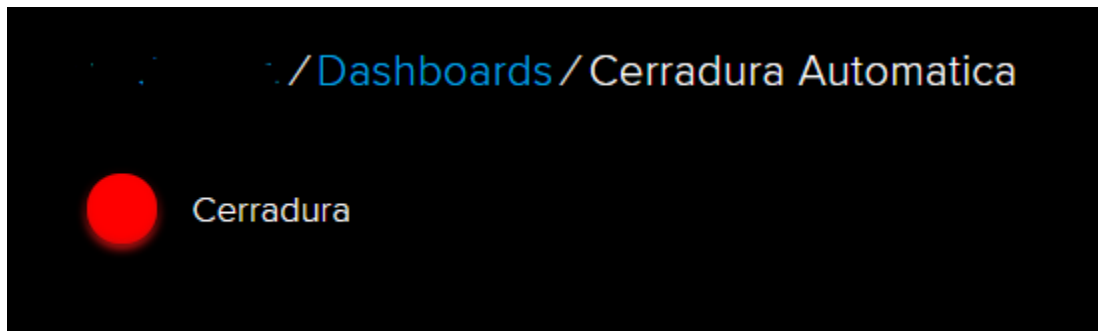
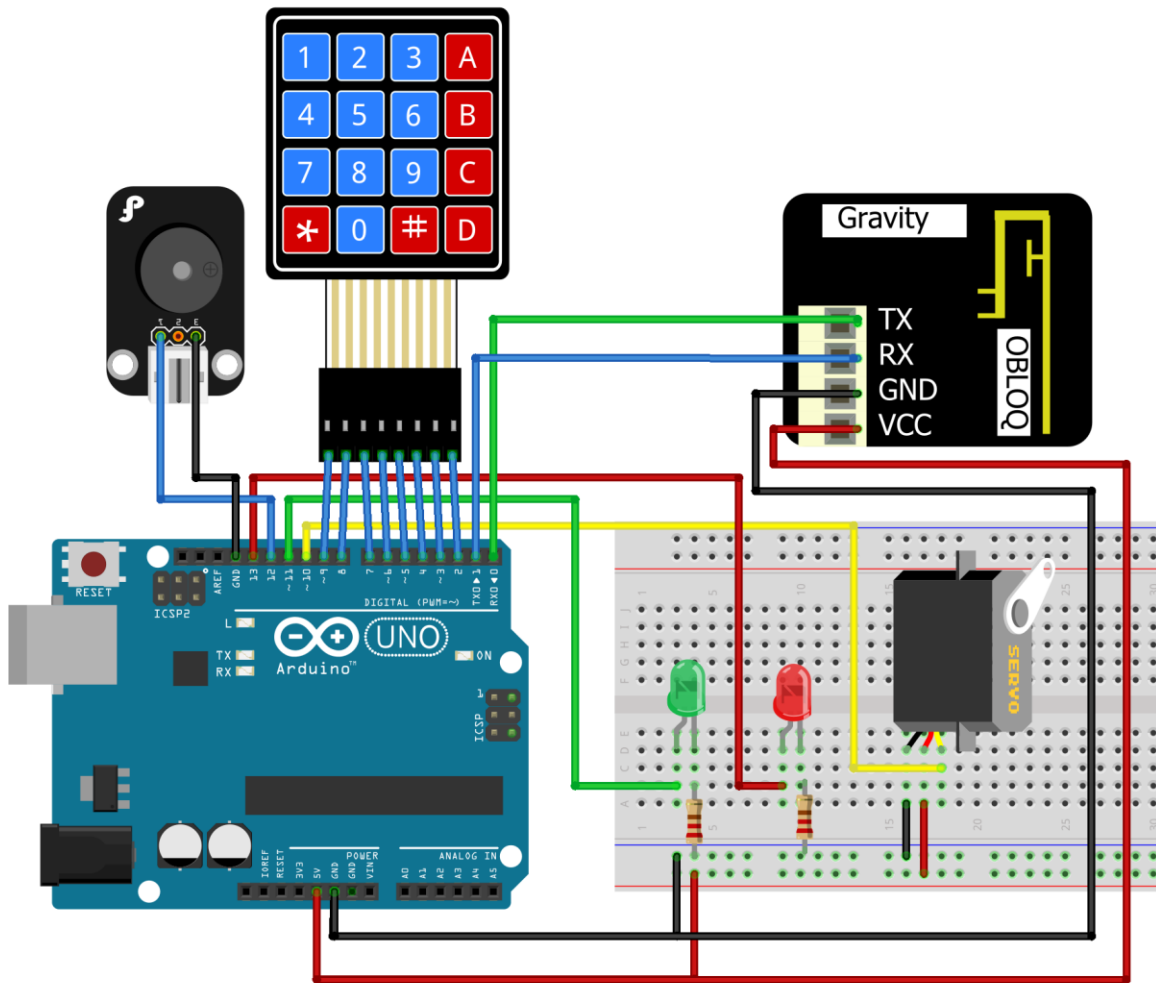


Fig.23

Una vez realizado el Panel, publicaremos el estado de la cerradura para monitorearlo de manera remota. Minimice la pantalla, luego sacaremos información de ella.

Paso 3 - Conectar módulo Obloq

A continuación se muestra el diagrama de conexión de Arduino UNO y OBLOQ. Para simplificar el proceso y la explicación, se implementará IoT en el código y circuito del nivel intermedio.



Esquema 5

Paso 4 - Arduino IDE

La programación por bloques tiene sus ventajas desde un punto de vista didáctico pero cuando el programa crece en complejidad puede resultar poco práctico. A menudo podemos encontrarnos con el hecho de que ciertas operaciones no pueden resolverse utilizando bloques o que hacerlo con este método resulta más engorroso y difícil de interpretar que si se utilizara el código escrito.

Hasta ahora hemos visto cómo al realizar nuestra programación en bloques se generaba simultáneamente un código escrito en el área lateral derecha. Para esta sección de la actividad se propone trabajar directamente sobre el código, para ello vamos a recurrir al entorno nativo de Arduino que llamamos Arduino IDE (entorno de desarrollo integrado).

Agregaremos, sobre el código realizado en el nivel intermedio, una nueva parte del programa que determinará la publicación del cambio de estado de la cerradura cada vez que esta se abra o se cierre.

Para ello, debemos agregar la siguiente línea, que informará cuando la cerradura esté abierta:

```
olq.publish("cerradura", 1); // Informa Cerradura Abierta
```

Y esta que indicará cuando la cerradura esté cerrada:

```
olq.publish("cerradura", 0); // Informa Cerradura Cerrada
```

Paso 6 - Programación IoT

Utilizaremos la librería ObloqAdafruit para informar a Adafruit cada vez que cambie el estado del semáforo. Podremos monitorear este estado desde el Panel de Control que hemos creado.

En primer lugar debemos instalar la librería en el Arduino IDE. Para esto debemos ingresar al menú Programa > Incluir Librería > Gestionar Librerías.

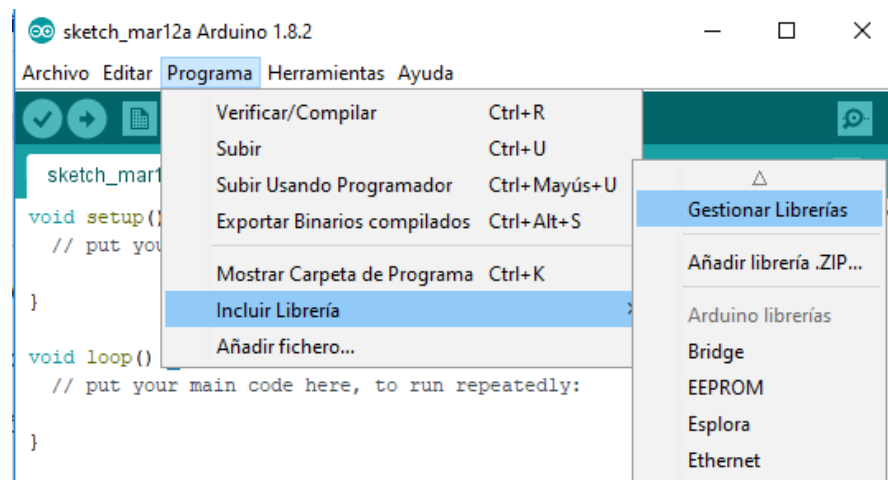


Fig. 24

Se abrirá una ventana con un buscador en margen superior. Debemos escribir Obloq, seleccionar la librería ObloqAdafruit y apretar el botón Instalar.

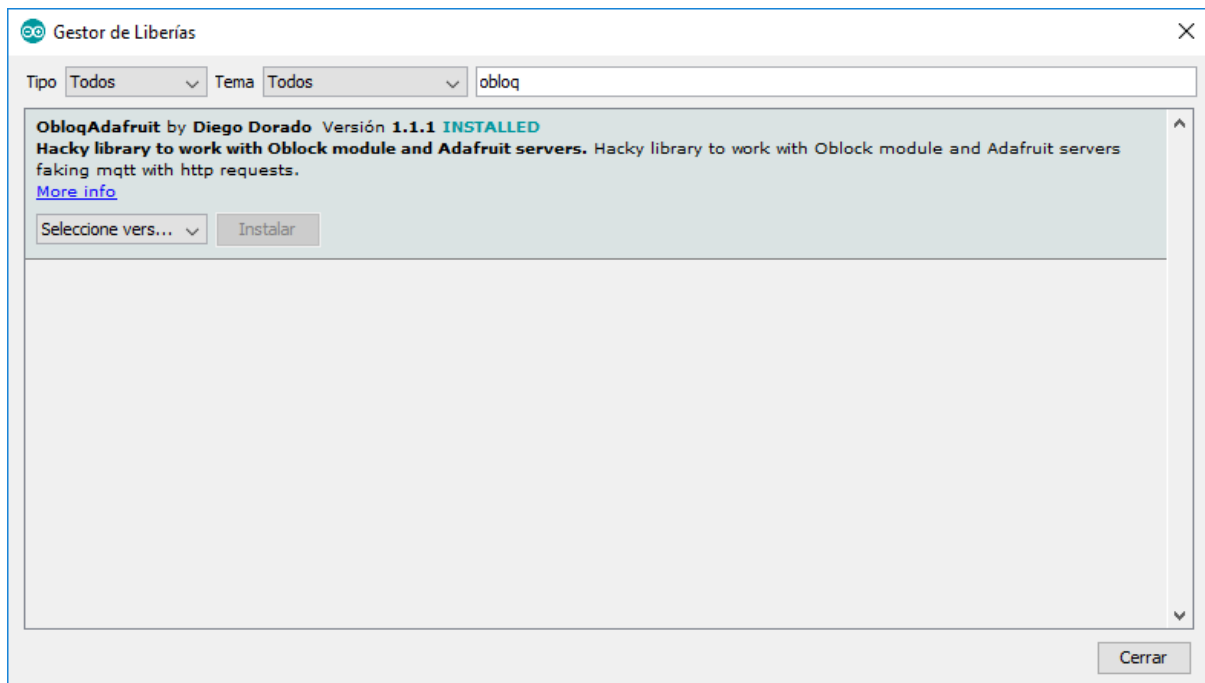


Fig. 25

En general las librerías traen códigos de ejemplo como referencia. Abrimos el ejemplo “Publicar” ubicado en Archivo > Ejemplos > ObloqAdafruit > Publicar.

Debemos reemplazar el SSID de la WiFi, su password, el IO_USERNAME e IO_KEY que son nuestras credenciales que copiaremos de Adafruit. Para ello maximizamos la pantalla de Adafruit y hacemos click en el ícono de la “llave”.

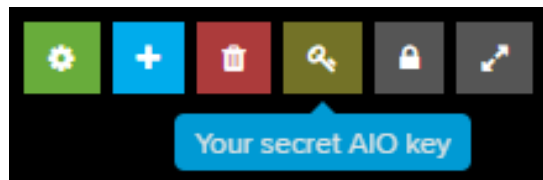


Fig. 28

Copiamos el código que nos ofrece para Arduino, con nuestro usuario y key.

YOUR AIO KEY

Your Adafruit IO key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your AIO key can view all of your data, create new feeds for your account, and manipulate your active feeds.

If you need to regenerate a new AIO key, all of your existing programs and scripts will need to be manually changed to the new key.

Username

usuario_aio

Active Key

1234cfdd29a244b6b049abb07727c117

REGENERATE AIO KEY

[Hide Code Samples](#)

Arduino

```
#define IO_USERNAME "usuario_aio"
#define IO_KEY      "1234cfdd29a244b6b049abb07727c117"
```

Fig. 29

Estos datos aparecen en el código de la siguiente manera:

```
#define IO_USERNAME "usuario_adafruit"
#define IO_KEY      "key_adafruit"
```

Se deberán reemplazar en esas dos líneas el usuario y key por los que se hayan obtenido en Adafruit. Por ejemplo:

```
#define IO_USERNAME "usuario_aio"
#define IO_KEY      "1234cfdd29a244b6b049abb07727c117"
```

También modificaremos `softSerial(10,11)` por `softSerial(8,9)` ya que así es como lo conectamos en nuestra placa, quedándonos el siguiente código:

```
#include "SoftwareSerial.h"
#include "ObloqAdafruit.h"

// Indicamos conexión de wifi.
#define WIFI_SSID      "SSID_de_Wifi"
#define WIFI_PASSWORD  "PWD_de_WIFI"

// Copiamos las credenciales obtenidas anteriormente en Adafruit.
```

```
#define IO_USERNAME    "usuario_adafruit"
#define IO_KEY         "key_adafruit"

SoftwareSerial softSerial(8,9);
ObloqAdafruit olq(&softSerial,WIFI_SSID,WIFI_PASSWORD,IO_USERNAME,IO_KEY);
```

El código quedará, entonces, como se ve a continuación:

```
#include <Keypad.h>
#include <Servo.h>
#include "SoftwareSerial.h"
#include "ObloqAdafruit.h"

// Indicamos conexión de wifi.
#define WIFI_SSID      "SSID_de_Wifi"
#define WIFI_PASSWORD  "PWD_de_WIFI"

// Copiamos las credenciales obtenidas anteriormente en Adafruit.
#define IO_USERNAME    "usuario_adafruit"
#define IO_KEY         "key_adafruit"

SoftwareSerial softSerial(0,1);
ObloqAdafruit olq(&softSerial,WIFI_SSID,WIFI_PASSWORD,IO_USERNAME,IO_KEY);

Servo Cerradura; // creamos el objeto servo que controlará la cerradura
const byte Filas = 4; //4 Filas.
const byte Columnas = 4; //4 Columnas.
//Definición de teclas del teclado.
char Teclas[Filas][Columnas] = {
    {'1','2','3','A'},
    {'4','5','6','B'},
    {'7','8','9','C'},
    {'*','0','#','D'}
};

byte Pins_Filas[Filas] = {9, 8, 7, 6}; //Conectar los pines de las filas
del teclado.
byte Pins_Columnas[Columnas] = {5, 4, 3, 2}; //Conectar los pines de las
columnas del teclado.
```

```

//Se inicia el teclado.
Keypad Teclado = Keypad(makeKeymap(Teclas), Pins_Filas, Pins_Columnas,
Filas, Columnas);

void setup(){
  pinMode(11,OUTPUT); //Configura el pin 11 como salida LED VERDE.
  pinMode(12,OUTPUT); //Configura el pin 12 como salida BUZZER.
  pinMode(13,OUTPUT); //Configura el pin 13 como salida LED ROJO.
  Cerradura.attach(10);
  Cerradura.write(0); //Inicia la apertura de la cerradura.
  digitalWrite(11, HIGH); //Enciende LED VERDE.
  softSerial.begin(9600);
}

int Clave[4]={'1','5','9','D'}; // Clave para acceder a la puerta.
int Clave_Pres[4]={'0','0','0','0'};
int indice=0;
int estado_puerta=1; // Inicialmente la puerta está abierta.
void loop(){
  char Tecla_Pres = Teclado.getKey(); //Toma el valor del teclado.

  if (Tecla_Pres){
    Clave_Pres[indice]=Tecla_Pres; //Guarda la tecla presionada.
    indice++;
    if(indice==4){ //Si se presionan 4 teclas se comparan las claves.
      indice=0;
      //Si las claves son iguales,
      if(Clave[0]==Clave_Pres[0] && Clave[1]==Clave_Pres[1] &&
Clave[2]==Clave_Pres[2] && Clave[3]==Clave_Pres[3]){
        digitalWrite(12, HIGH); //enciende el buzzer.
        if(estado_puerta==0){
          Cerradura.write(0); //Abre el mecanismo de la cerradura.
          estado_puerta=1; //Estado de puerta Abierta.
          digitalWrite(11, HIGH); //Enciende LED VERDE
          digitalWrite(13, LOW); //Apaga LED ROJO.
          olq.publish("cerradura", 1); // Informa Cerradura Abierta.
        }else{
          Cerradura.write(90); //Cierra el mecanismo de la cerradura.
          estado_puerta=0; //Estado de puerta cerrada.
          digitalWrite(11, LOW); //Apaga LED VERDE.
          digitalWrite(13, HIGH); //Enciende LED ROJO.
          olq.publish("cerradura", 0); // Informa Cerradura Abierta.
        }
      }
    }
  }
}

```

```
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer. }else{//Código
ingresado incorrecto, emite doble pitido.

        digitalWrite(12, HIGH); //Enciende el buzzer.
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer.
        delay(300);
        digitalWrite(12, HIGH); //Enciende el buzzer.
        delay(200);
        digitalWrite(12, LOW); //Apaga el buzzer.

    }
}

olq.update();
}
```

Cierre

Una vez finalizado este proyecto, es posible extenderlo si se quiere continuar. Estas son algunas opciones sugeridas:

- Incluir una llave y un lector RFID
- Abrir y cerrar la cerradura de manera remota a través de IoT.

El proceso de resolución de problemas como los que se han planteado aquí permite la movilización y la integración de distintos saberes en la búsqueda de soluciones posibles a una situación dada. Si bien la información aquí fue presentada a modo de instructivo, se espera que sean los estudiantes organizados en pequeños grupos quienes vayan encontrando las mejores formas para construir los dispositivos. Esto implica preparar los materiales para que cada grupo cuente con todo lo necesario para la construcción del proyecto. Además, al interior de cada grupo, los estudiantes deben distribuirse los roles y las tareas de acuerdo a las demandas que van teniendo en las actividades.

Es importante que los docentes acompañen las producciones de cada grupo monitoreando los avances de todos los estudiantes y presentando la información que se considere necesaria para continuar la tarea. Pero, al mismo tiempo, es necesario que habiliten espacios para que los alumnos realicen hipótesis, planteen interrogantes, indaguen, prueben y realicen ajustes de acuerdo a lo que ellos mismo van pensando sobre cómo llevar a cabo el proyecto.

En este sentido, registrar lo que se va haciendo, las preguntas de los alumnos, las pruebas, los errores y cómo se fueron construyendo los dispositivos, permite reflexionar sobre la propia práctica, reforzar los aprendizajes construidos a lo largo de este proceso y poder volver a ese material disponible para próximos proyectos que se realicen.

Una vez terminado el proyecto, se sugiere reunir y organizar con el grupo el registro que se hizo del proceso realizado. Esta instancia de sistematización también permite movilizar capacidades vinculadas a la comunicación porque implica tomar decisiones respecto a cómo se quiere mostrar el proyecto a otros (otros grupos, otras escuelas, otros docentes, a la comunidad, etc.).

Glosario

Electrónica y Arduino

Arduino: Placa electrónica que contiene un microcontrolador programable y sistema de comunicación (USB y serial) que permite al usuario cargarle diversos programas así como también comunicarse con la misma. Del lado de la computadora se utiliza un IDE de programación para generar el código, compilarlo y quemarlo en la placa. Existen múltiples IDE compatibles con las placas Arduino.

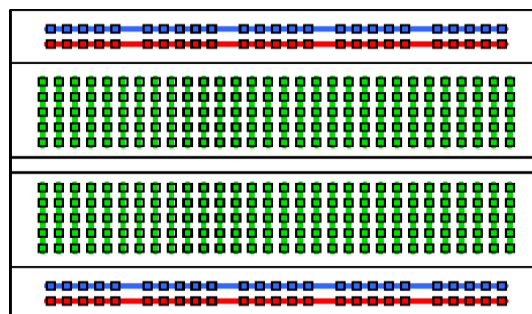
El microcontrolador posee entradas analógicas y digitales así como salidas digitales, PWM y servo. Las entradas y salidas digitales son las que permiten leer o escribir estados del tipo binarios. Pueden adoptar la forma de 0 ó 1, alto o bajo, verdadero o falso. Para prender y apagar los LED del semáforo utilizamos salidas digitales, las mismas están nombradas con números desde el 0 al 13.

Las entradas analógicas permiten leer información que puede adoptar diferentes niveles de tensión, tal como la lectura de un termómetro analógico, la posición de un potenciómetro, etc. Las mismas están identificadas en la placa como A0 a A5.

Puerto COM: Es el puerto de comunicaciones a través del cual un sistema operativo informático se comunica con un dispositivo externo tal como una placa Arduino. La asignación de los mismos suele realizarse de forma automática al conectar la placa via USB. Dicha asignación suele ser dinámica, lo que significa que a veces cambia el número al conectar una misma placa en otro puerto USB o al conectar varias placas. En todos los IDE de programación es necesario especificar el puerto COM a través del cual nos comunicaremos con la placa Arduino.

Protoboard: Es una placa experimental que permite el prototipado rápido de circuitos electrónicos. Tiene orificios para insertar las patas de los componentes permitiendo que se conecten sin tener que recurrir a la soldadura.

El mismo posee una grilla de orificios que se encuentran conectados entre sí siguiendo el esquema de la imagen. Las líneas de conexión superior e inferior recorren la placa de punta a punta y suelen utilizarse para la alimentación del circuito, mientras que las líneas verdes se suelen utilizar para interconectar componentes. Tomar en cuenta que las líneas verdes se interrumpen en el centro de la placa. Generalmente se utilizan cables del tipo dupont para realizar conexiones en la protoboard



Servomotor: Un servo es un dispositivo que se compone de un motor y un sistema de control que le permite ubicarse en una posición específica. Los servos más comunes pueden moverse en un rango de 0° a 180°, sin poder girar de forma continua. Se suelen utilizar en aplicaciones tipo barreras o brazos mecánicos. La programación de los mismos es muy simple, teniendo que especificar únicamente el ángulo al que se lo quiere posicionar.

Existen también servos de “rotación continua” que permiten realizar un control relativamente preciso del movimiento así como también que el eje de giros continuos sin estar acotado a un rango de movimiento como el caso de los servos estándares. Este tipo de servo requiere una lógica de programación un poco más compleja que el caso anterior.

Los servos tienen 3 pines de conexión, dos de ellos se ocupan en alimentación eléctrica (VCC marrón y GND negro) y un tercer pin que se conecta a una salida digital del Arduino (cable naranja). Para controlar el servo el Arduino genera una señal con una frecuencia particular y un método de modulación de pulsos cuyo ciclo de trabajo equivale a el ángulo que se desea posicionar el servo, no confundir este método de modulación con el PWM.

LED: Componente electrónico tipo diodo que emite luz. Es necesario tomar en cuenta la polaridad del mismo para ponerlo en funcionamiento. Conectándolo con la polaridad invertida generalmente no va a traer mayores consecuencias que la imposibilidad de hacer que encienda. Existen dos formas de distinguir la polaridad del mismo: podemos identificar la pata negativa como la pata más corta u observando el lado plano en el encapsulado del mismo.



Resistencia: La resistencia eléctrica es una característica de todo material conductor eléctrico de hacer oposición al paso de la corriente eléctrica, es uno de los componentes más utilizados en la electrónica. El valor resistivo se mide en ohms y se representa con el símbolo Ω . Existen resistencias de valores que van desde menos de 1 ohm hasta varios millones. Se suelen utilizar para determinar la cantidad de corriente de una rama de circuito, por ejemplo para evitar que se queme el LED por exceso de corriente.

Pulsador: Es un interruptor eléctrico que se encuentra normalmente abierto y al presionarlo se cierra el circuito entre sus patas. Se suele usar como botón para que el usuario pueda interactuar con un dispositivo así como también como sensor para detectar cuando un móvil llegó a toparse con un obstáculo (sensor de choque o sensor de fin de carrera).

Teclado de membrana: El teclado de membrana está compuesto por dos láminas flexibles con circuitos impresos sobre una tinta conductora. Al presionar una tecla, las membranas hacen contacto cerrando el circuito. El diseño eléctrico del teclado es matricial, lo que significa que hay pines que representan cada fila así como también cada columna. La pulsación de cada tecla equivale a un cruce entre una fila y una columna. Se suele utilizar una librería de Arduino que facilita el proceso de lectura que permite identificar la tecla presionada.

Buzzer: Un **Zumbador** (*buzzer*) es un dispositivo que genera sonido con una frecuencia determinada. Al recibir alimentación eléctrica comienza a sonar, controlando la misma se pueden generar sonidos continuos o pulsos. Sirve como mecanismo de señalización o aviso.

Si se quisiera poder tener control de la frecuencia y nivel del sonido generado se debería utilizar otro tipo de dispositivo, tal como un parlante pequeño.

Impresión 3D

Formato .stl: El .stl es un formato de archivo que contiene la forma de un objeto sólido. Este formato de archivo no contiene información tal como color, texturas o propiedades físicas. Los archivos STL son objetos ya consolidados por lo que resulta útil para el intercambio e impresión de los mismos. Este formato de archivo no resulta práctico en caso de necesitar editar la geometría del objeto. Esto se debe a que no contiene información paramétrica sobre la generación de las diversas formas, curvas o capas que se utilizan a la hora de diseñar. Se lo puede considerar como la bajada o exportación final de un diseño, en ciertos aspectos equivalente a pensar en exportar un documento PDF partir de un documento de texto. Es posible generar archivos STL partiendo desde distintos tipos de entornos de diseño y modelado en 3D.

Código G: Es el nombre que recibe el conjunto de acciones que va a tener que realizar la impresora 3D, o cualquier otro tipo de máquina CNC, para completar el trabajo solicitado. Estas instrucciones se generan partiendo del análisis de un archivo STL y realizando el cálculo de todos los movimientos y trayectorias que realizará cada componente de la impresora (motores, avance de filamento, calentador de extrusor, calentador de la base, etc) para materializar el objeto en cuestión. Debido a que cada marca y modelo de impresora 3D tiene diferentes características mecánicas, el código G generado para imprimir cierto objeto solo va a servir para ejecutarse en un modelo de impresora específico.

Internet de las cosas

Panel de Control Adafruit: Los sistemas IoT trabajan apoyándose en un servidor que se encarga de centralizar y gestionar la información que reportan los diversos sensores así como responder a las consultas de los dispositivos que buscan acceder a dicha información (mostrarla en pantalla, tomar decisiones, etc). Adafruit es una plataforma online con posibilidad de uso gratuito que ofrece el servicio de gestión de esta información. La misma ofrece un alto grado de compatibilidad con diversos estándares de trabajo IoT y se encuentra principalmente orientada al uso educativo.

Feed: fuente de datos en la que uno puede publicar y a la que puede suscribirse. Es decir, permite enviar datos, para que estos sean almacenados en el tiempo así como también leerlos, recibiendo las actualizaciones de quienes estén publicando allí. Es una forma de almacenar información en una gran base de datos de forma ordenada, utilizando el concepto de etiquetas tanto al momento de escribirla como el de leerla.

Reconocimientos

Este trabajo es fruto del esfuerzo creativo de un enorme equipo de entusiastas y visionarios de la pedagogía de la innovación, la formación docente, la robótica, la programación, el diseño y la impresión 3D. Les agradecemos por el trabajo en equipo inspirador para traer a la realidad la obra que, en forma conjunta, realizamos INET y EDUCAR del Ministerio de Educación, Cultura, Ciencia y Tecnología de la Nación Argentina.

Contenidos

Equipo INET

Alejandro Anchava
Joreliz Andreyana Aguilera Barragán
Omar Leandro Bobrow
Alejandro Cesar Cáceres
Ezequiel Luberto
Gustavo Roberto Mesiti
Alejandro Palestrini
Judit Schneider
Pablo Trangone

Equipo Educar:

Pablo Aristide
Mayra Botta
Anabela Cathcarth
Eduardo Chiarella
María Laura Costilla
Diego Dorado
Facundo Dyszel
Federico Frydman
Matías Rinaldi
Uriel Rubilar
Camila Stecher
Carolina Sokolowicz
Nicolás Uccello

Para la confección de esta obra se contó con el apoyo de la Universidad Pedagógica Nacional "UNIPE". En particular en el desarrollo de los capítulos 1 y 2, los cuales estuvieron a cargo de los profesores Fernando Raúl Alfredo Bordinon y Alejandro Adrián Iglesias.

Producción y comunicación

Juliana Zugasti

Diseño y edición

Leonardo Frino
Mario Marrazzo

Corrección de estilo

María Cecilia Alegre

Agradecimientos especiales

Mariano Consalvo. Equipo ABP

Damián Olive. Equipo de ABP

María José Licio Rinaldi, Directora Nacional de Asuntos Federales INET, quien siempre acompañó a este equipo en todas las gestiones para su implementación

Estamos comprometidos en instalar la innovación en la escuela secundaria técnica: la robótica, la programación, el pensamiento computacional, los proyectos tecnológicos, el ABP, la impresión 3D, de manera más accesible para todos.

Agradecemos enormemente, docente, tu continua dedicación y compromiso con el futuro de tus estudiantes.

¡Estamos ansiosos por saber qué es lo que vamos a crear juntos!