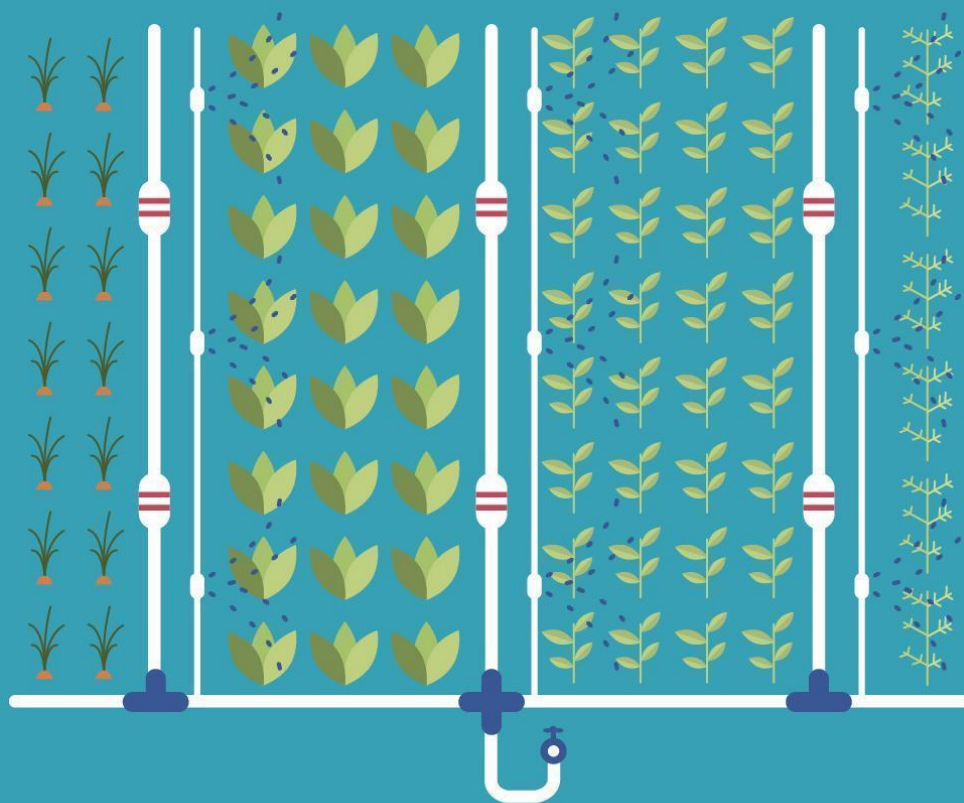


# SABERES DIGITALES



# 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

## SISTEMA DE RIEGO AUTOMATIZADO

Ficha técnica	3
Presentación	4
<b>Nivel Inicial</b>	<b>5</b>
<b>Paso 1 - Conectar la electroválvula</b>	<b>5</b>
<b>Paso 2: Programar el funcionamiento del dispositivo</b>	<b>7</b>
<b>Paso 3: Determinar tiempos de riego adecuados</b>	<b>10</b>
<b>Nivel Intermedio</b>	<b>12</b>
<b>Paso 1 - Conectar el higrómetro</b>	<b>12</b>
<b>Paso 2 - Leer el puerto serie</b>	<b>14</b>
<b>Paso 3 - Leer el higrómetro por el puerto serie</b>	<b>16</b>
<b>Paso 4 - Determinar los valores de humedad para activar la válvula</b>	<b>18</b>
<b>Paso 5 - Programar la apertura y cierre de la válvula en base a la humedad del suelo</b>	<b>18</b>
<b>Nivel Avanzado</b>	<b>20</b>
<b>Paso 1 - Introducción a Internet de las Cosas (IoT)</b>	<b>21</b>
<b>Paso 2 - Crear un Panel de Control</b>	<b>21</b>
<b>Paso 3 - Agregar módulo OBLOQ</b>	<b>30</b>
<b>Paso 4 - Arduino IDE</b>	<b>30</b>
<b>Paso 5 - Programar sin código bloqueante</b>	<b>32</b>
<b>Paso 6 - Programación IoT</b>	<b>34</b>
<b>Cierre</b>	<b>39</b>
Glosario	39
Reconocimientos	42

# SISTEMA DE RIEGO AUTOMATIZADO

## Ficha técnica

<b>Nivel educativo</b>	Secundario. Ciclo Básico.
<b>Descripción Gral</b>	Diseño, construcción y programación de un sistema de riego automatizado.
<b>Niveles de complejidad</b>	<p>Nivel inicial: instalar un sistema de riego que funcione mediante una <b>Placa Arduino</b> y esté programado para dejar pasar el agua por intervalos definidos de tiempo.</p> <p>Nivel intermedio: agregar <b>sensores de humedad</b> que permitan regar sólo cuando sea necesario.</p> <p>Nivel avanzado: informar los datos obtenidos a un dispositivo móvil a través de <b>internet de las cosas (IoT)</b>.</p>
<b>Insumos</b>	<ul style="list-style-type: none"><li>• 1 x Módulo relé de 5V</li><li>• 1 x Módulo Reloj</li><li>• 1 x Higrómetro de suelo</li><li>• 1 x Electroválvula de lavarropas 180°</li><li>• Cables dupont macho hembra</li><li>• Resistencias</li><li>• Mangueras</li><li>• 1 x Arduino UNO R3</li><li>• 1 x Cable UBS tipo B</li><li>• 1 x Fuente de 9V 1 A (plug centro positivo, 5.5 x 2.1 mm)</li></ul>
<b>Equipamiento</b>	<ul style="list-style-type: none"><li>• Computadora</li><li>• Soldador</li><li>• Estaño</li><li>• Alicates</li><li>• Pinza de punta</li><li>• Brusela</li></ul>
<b>Otros requisitos</b>	<ul style="list-style-type: none"><li>• Conexión a internet</li><li>• Descargar el programa "mBlock3"</li></ul> <p><a href="http://www.mblock.cc/software-1/mblock/mblock3/">http://www.mblock.cc/software-1/mblock/mblock3/</a></p>

# Presentación

## **Descripción ampliada del proyecto**

En este proyecto, se propone instalar un sistema de riego automatizado utilizando una placa Arduino, una electroválvula y un sistema de mangueras. Para implementar este sistema no es necesario contar con una huerta, es posible utilizarlo para regar plantas en macetas y canteros.

Inicialmente, el sistema será programado para regular el paso de agua con una electroválvula, estableciendo intervalos de riego con una duración previamente determinada por el usuario.

En el nivel de complejidad intermedio, se propone incluir un higrómetro que mida la humedad de la tierra, y regule la apertura de la electroválvula a partir de dicha información. En el nivel más avanzado, se propone incorporar Internet de la Cosas (IoT) para monitorear, de manera remota, el nivel de humedad de suelo y el estado de la electroválvula.

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 al manejo de distintos componentes electrónicos mediante la construcción de un sistema de riego automatizado.
- Introducirse en el armado de circuitos utilizando placas Arduino, sensores y actuadores.
- Analizar y desarrollar la programación de la estructura secuencial de un programa que permita regular el tiempo de riego de manera automática.
- Utilizar un higrómetro para medir la humedad del suelo y regular el tiempo de riego en base a la información obtenida (nivel intermedio).
- Utilizar Internet de la Cosas (IoT) para registrar y monitorear la humedad del suelo.

## Desarrollo

### Nivel Inicial

En una escuela, se les propuso a los estudiantes de segundo año realizar una huerta en el patio. Los docentes y los alumnos prepararon el terreno y seleccionaron las plantas que incluirían en la huerta, en base a las condiciones del suelo y climáticas de la zona.

Uno de los temas fundamentales que debieron contemplar para el cuidado de su huerta fue el riego: debían definir su frecuencia y la cantidad de agua necesaria. El agua es un elemento esencial para la supervivencia de las plantas y la falta o el exceso de la misma podrían afectar su crecimiento.

Una vez definidas estas variables, los docentes les presentaron a los estudiantes el desafío de generar un sistema que riegue las plantas de manera automática, controlando la cantidad de agua y los intervalos de riego.

**En esta instancia del proyecto se propone la construcción de un sistema de riego automatizado, que regule el paso de agua por intervalos de tiempo previamente definidos, utilizando una electroválvula y un módulo reloj conectados a una placa Arduino. Este sistema de riego puede diseñarse y construirse para que los estudiantes aprendan a hacerlo aunque no se cuente, en el momento, con un espacio para armar una huerta. El dispositivo puede implementarse en una maceta o un cantero.**

### Paso 1 - Conectar la electroválvula

El paso del agua a la manguera del sistema de riego es regulado por una electroválvula (una válvula que se controla eléctricamente, como las que se utilizan habitualmente en los lavarropas). La válvula se mueve mediante una bobina solenoide que permite el paso del agua al recibir corriente alterna de 220V en sus bornes.

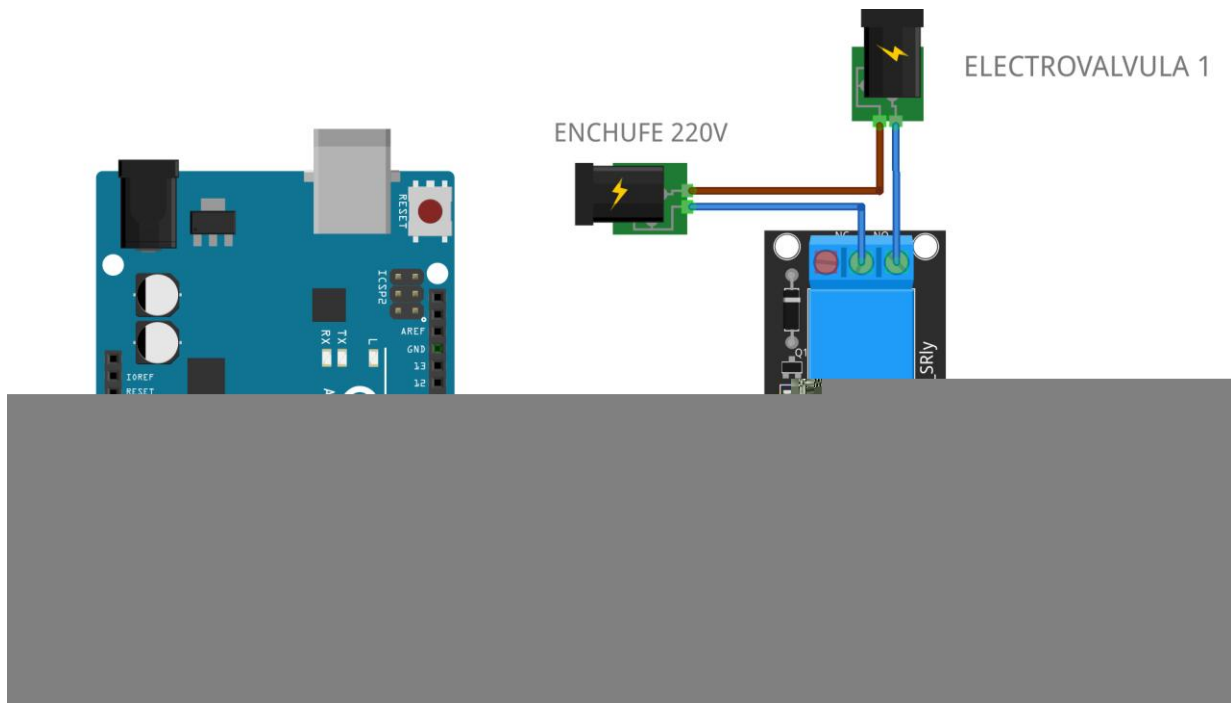


Electroválvula, es un actuador que permite o interrumpe el paso de un fluido a través del mismo. Existen versiones que trabajan con 12 V así como también los que trabajan directamente con 220 V, permitiendo trabajar en un sistema de mayor nivel de caudal

La electroválvula tiene dos conectores para manguera (uno para la entrada y otro para la salida de agua) y dos terminales eléctricos tipo pala. En este proyecto conectaremos la electroválvula a un módulo relé, que se conectará a la placa Arduino para poder controlarlo electrónicamente.

Lo primero que haremos, entonces, será instalar las mangueras correspondientes de entrada y salida a la electroválvula. La manguera de entrada irá conectada a una canilla y la de salida será la que lleve el agua directamente a las plantas.

Para poder activar y desactivar la válvula la conectaremos como indica el siguiente esquema:



Esquema 1

**¡Atención! Para construir este dispositivo trabajaremos con una tensión de 220 V. Si se está trabajando con protoboard, se recomienda no incluir en el mismo las conexiones a relé y 220V.**

Como se puede ver en la imagen, utilizaremos un relé entre la toma de 220V y la electroválvula.

En la referencia “ELECTROVÁLVULA 1” se conectan los dos terminales de la electroválvula (pueden conectarse indistintamente a uno u otro terminal ya que no poseen polaridad). Y en la referencia “ENCHUFE 220V” se conecta a la red eléctrica.

El código que subamos a la placa Arduino controlará la apertura y el cierre del relé y, en consecuencia, la apertura y el cierre de la electroválvula.

## Paso 2: Programar el funcionamiento del dispositivo

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:

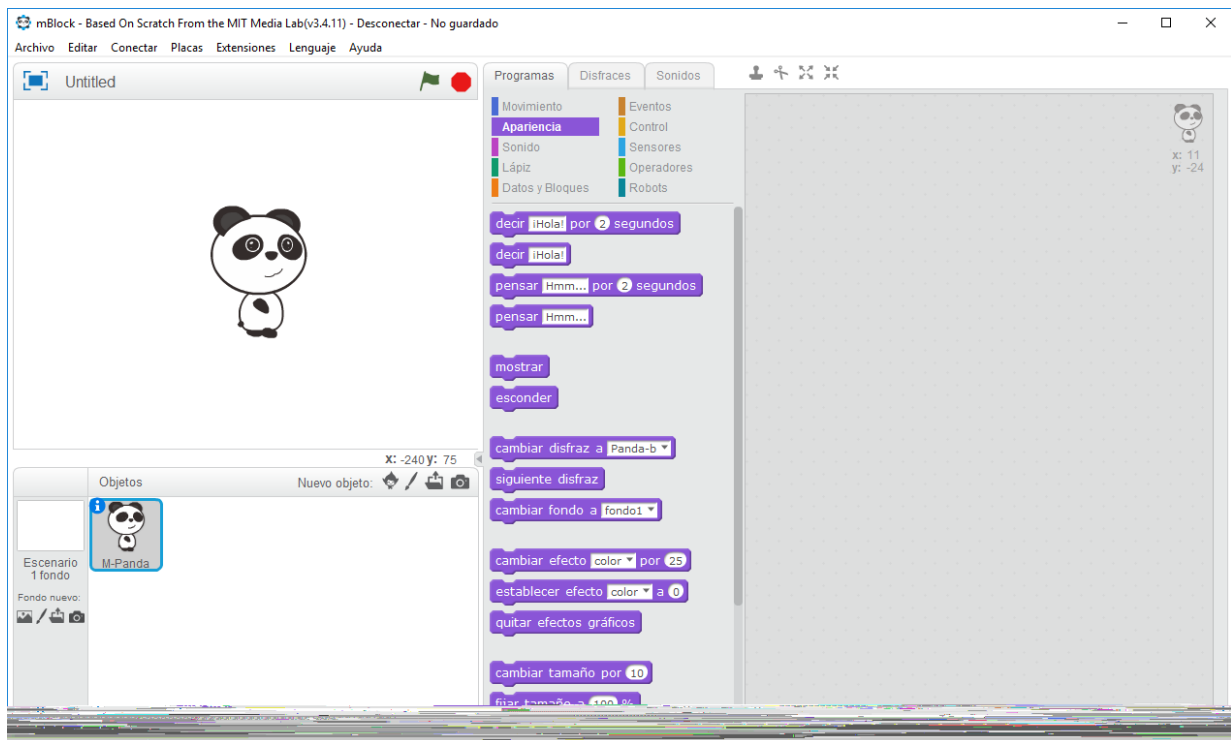


Fig. 1

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.

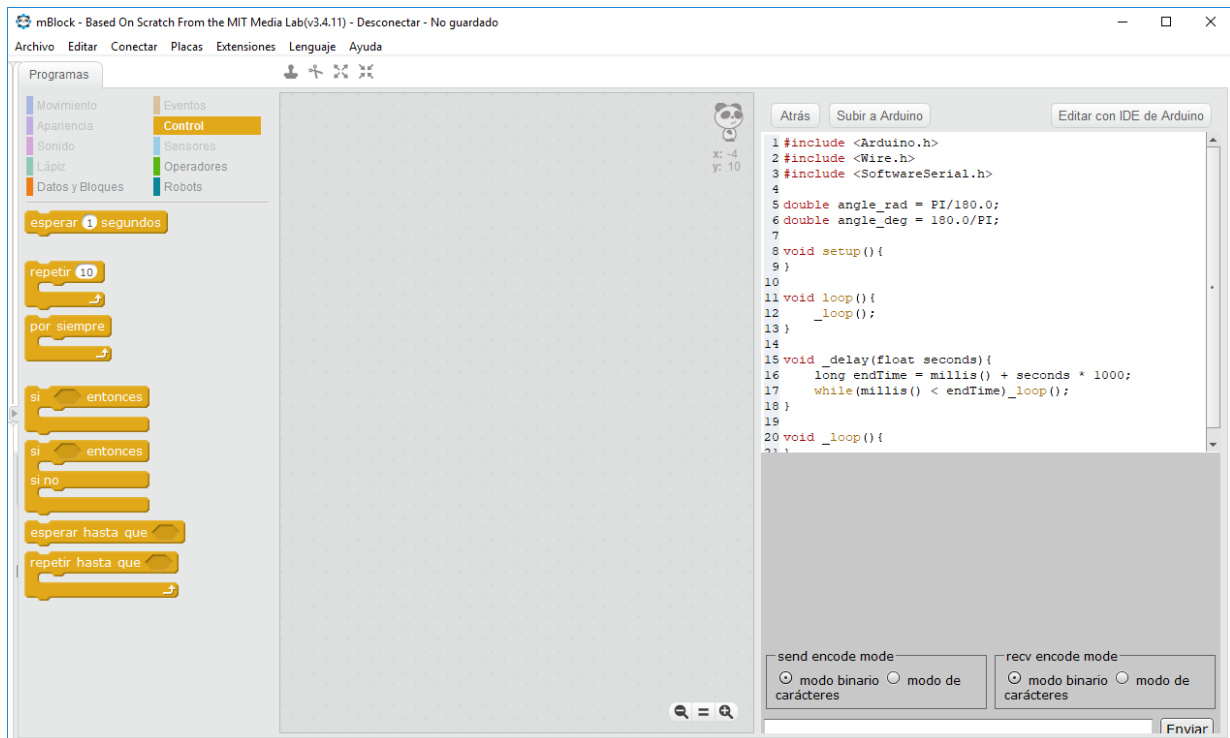


Fig. 3

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

Para asegurarnos que el sistema funcione correctamente, podemos hacer una prueba simple estableciendo un tiempo de apertura y otro de cierre para la electroválvula.

Cuando no recibe corriente eléctrica, la electroválvula está normalmente cerrada. Escribimos un programa que mantenga la electroválvula abierta por diez segundos y luego

la cierre por otros diez segundos. Antes de probar el sistema, hay que asegurarse de que la canilla (o la salida de agua corriente) esté abierta.

El programa queda como sigue:



Fig. 5

Al asignar el estado “ALTO” a la salida del relé (“pin digital 2”) estamos indicando que se envíe una corriente de 5V para el relé que permita el paso de corriente a la electroválvula para que se abra.

Al asignarle el estado “BAJO”, estamos indicando que no se envíe corriente al módulo relé para que no permita el paso de corriente a la electroválvula, de manera que la misma se cierre.

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>

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

void setup(){
    pinMode(2,OUTPUT);
}

void loop(){
    digitalWrite(2,1);
    _delay(10);
    digitalWrite(2,0);
    _delay(10);
    _loop();
}
```

```

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

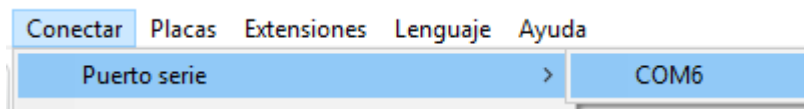
void _loop(){
}

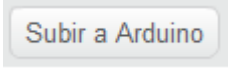
```

Fragmento de código 1

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  que se encuentra en la parte derecha superior de la interfaz.

Al terminar de subir nuestro código, veremos este mensaje

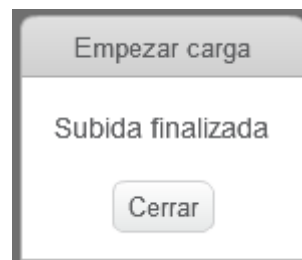


Fig. 6

### Paso 3: Determinar tiempos de riego adecuados

En el programa anterior, la válvula permanecía abierta por diez segundos y luego cerrada por otros diez segundos, a modo de ejemplo.

La frecuencia de riego necesaria variará según la necesidad de las plantas seleccionadas, las condiciones climáticas, el tipo de suelo y su cobertura (por ejemplo, si tiene hojarasca o no), y la cantidad de agua aportada en cada evento de riego. En este sentido, sería interesante que los estudiantes investiguen previamente acerca de las necesidades de las plantas seleccionadas, mediante la búsqueda de información en distintas fuentes o la realización de entrevistas a expertos sobre el tema. En el nivel de complejidad intermedio de este proyecto, se propone determinar la frecuencia de riego con el uso de un higrómetro (medidor de humedad).

Una vez determinado el tiempo de riego más adecuado para nuestra huerta, tendremos que modificar el programa. Este tiempo tiene que estar expresado en segundos. En el siguiente ejemplo, la válvula permanecerá cerrada una hora (3600 segundos) y luego se abrirá dos minutos (120 segundos):

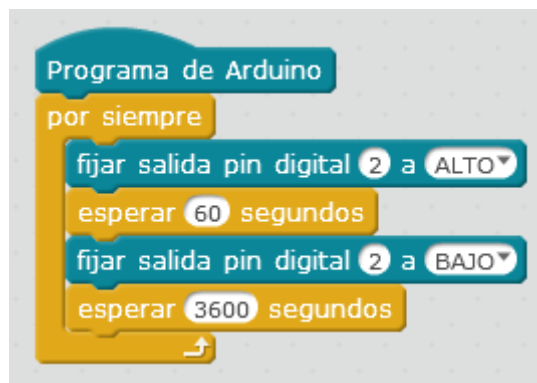


Fig. 7

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>

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

void setup(){
  pinMode(2,OUTPUT);
}

void loop(){
  digitalWrite(2,1);
  _delay(60);
  digitalWrite(2,0);
  _delay(3600);
}
```

```

    _loop();
}

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

void _loop(){
}

```

Fragmento de código 2

## Nivel Intermedio

Luego de haber utilizado este sistema de riego por varios meses, de realizar observaciones y registros de lo que sucede con sus plantas, los estudiantes notan que la frecuencia de riego necesaria varía en función de distintos factores que afectan a la humedad del suelo.

Uno de ellos es la temperatura ambiente que, al aumentar, genera que el agua que está en el suelo se evapore más rápido. Por ese motivo, deciden instalar un sensor de humedad del suelo que les permita saber cuándo es necesario regar y, así, evitar el consumo de agua cuando no se necesita.

**En esta instancia del proyecto se propone incluir al sistema un higrómetro de suelo (FC28), que permita medir la humedad del suelo y regular la apertura de la electroválvula en base a los datos obtenidos.**

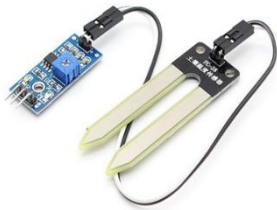
### Paso 1 - Conectar el higrómetro

En el nivel inicial de este proyecto, desarrollamos un programa que controle la frecuencia de riego de nuestro sistema, mediante la apertura y el cierre de una electroválvula conectada a dos mangueras. Ahora se propone ajustar la frecuencia de riego, en base a la información que obtengamos por medio de la medición de la humedad del suelo.

Podemos programar el sistema para que riegue cuando la humedad del suelo sea baja, y deje de regar cuando esta alcance un valor que consideramos suficiente. Para esto tendremos que definir un rango de valores que representen las condiciones de humedad del suelo. Nuestro sistema utilizará el higrómetro de suelo (FC28) para obtener los valores de humedad en cada momento. Se cotejarán con nuestro rango previamente definido y nos permitirá determinar si la válvula debe abrirse o cerrarse en función del nivel de humedad que registre.

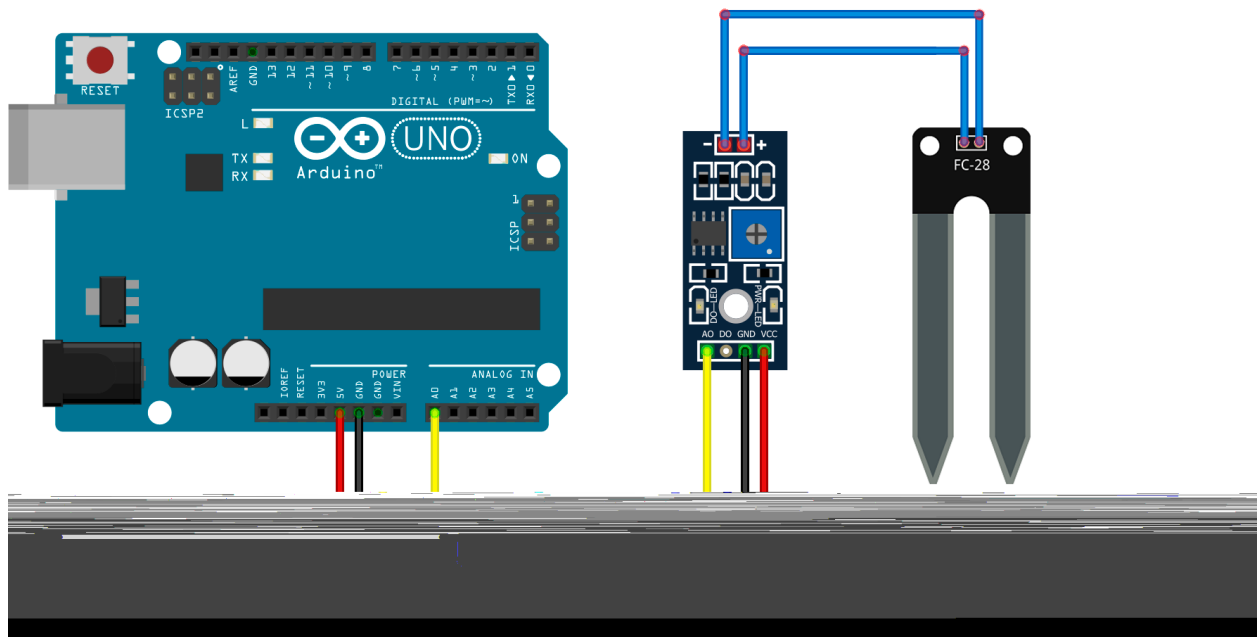
Más adelante explicaremos cómo leer y definir estos valores para el higrómetro. Para obtener una buena medición se debe enterrar la horquilla del sensor en un lugar que

represente lo más fielmente posible las condiciones generales de nuestra huerta, cantero o maceta. Es importante tener en cuenta que una mala ubicación del sensor (por ejemplo, en un espacio que se encuentre por fuera de la zona de riego) puede generar que algunas zonas queden muy secas y otras muy húmedas.



El higrómetro es un sensor que mide la humedad del suelo mediante la medición de la conductividad eléctrica.

Conectar el higrómetro a la placa Arduino como indica el siguiente esquema:



Esquema 2

Las entradas analógicas de la placa Arduino (ubicadas en los pines A0 al A5) digitalizan la señal conectada a la misma y devuelven un valor numérico en relación al nivel de tensión registrado. Los valores digitalizados pueden moverse en un rango de 0 a 1023. Siendo 1023 el equivalente a una tensión de entrada de 5V, 512 el equivalente a una tensión de entrada de 2,5V, etc

## Paso 2 - Leer el puerto serie

Cuando trabajamos con entradas digitales, que cuentan solamente con dos estados posibles, podemos reflejar su estado en cada momento con un dispositivo sencillo como, por ejemplo, un LED que se prenda o se apague. Pero cuando usamos pines analógicos, como el que utilizaremos para el higrómetro, no tenemos dos sino un rango de valores posibles (en este caso, todos los distintos niveles posibles de humedad de la tierra). Para ver esos datos y poder tomar decisiones en base a ellos podemos utilizar el “puerto serie”.

El puerto serie es un canal de comunicación entre la placa Arduino y la computadora. En este caso, lo usaremos para ver en la computadora los valores que registra el higrómetro, es decir, el valor leído en la entrada analógica A0 en donde conectamos el higrómetro.

En la placa Arduino el puerto serie está físicamente conectado al pin 0 (receptor) y al pin 1 (transmisor). Notaremos que en la serigrafía de la placa están identificados:

- RX <- 0
- TX -> 1

En proyectos donde utilicemos el puerto serie no debemos conectar nada a estos dos pines porque se utilizan para la comunicación. Estos dos pines están convenientemente vinculados al mismo cable USB que utilizamos para subir los programas a la placa Arduino. Es decir que este puerto serie nos sirve tanto para subir los programas a la placa, como para enviar mensajes entre la placa y la computadora.

Veamos un programa que simplemente nos envíe un mensaje de saludo cada 1 segundo. Escribimos un programa como el siguiente:

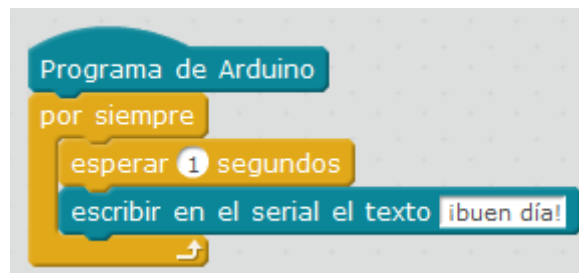


Fig. 8

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>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
```

```

void setup(){
    Serial.begin(115200);
}

void loop(){
    _delay(1);
    Serial.println("¡buen día!");
    _loop();
}

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

void _loop(){
}

```

Fragmento de código 3

Una vez subido el programa a la placa, tenemos que conectarnos al puerto serie para poder recibir estos mensajes. Esto lo hacemos seleccionando el puerto serie desde el menú.

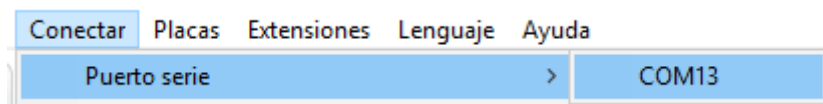


Fig. 9

En el área derecha, debajo del código escrito, comenzaremos a recibir mensajes ininteligibles cada 1 segundo. Para poder leer correctamente nuestros mensajes debemos cambiar el modo de codificación, que se encuentra seteado en “modo binario”, a “modo de caracteres”.

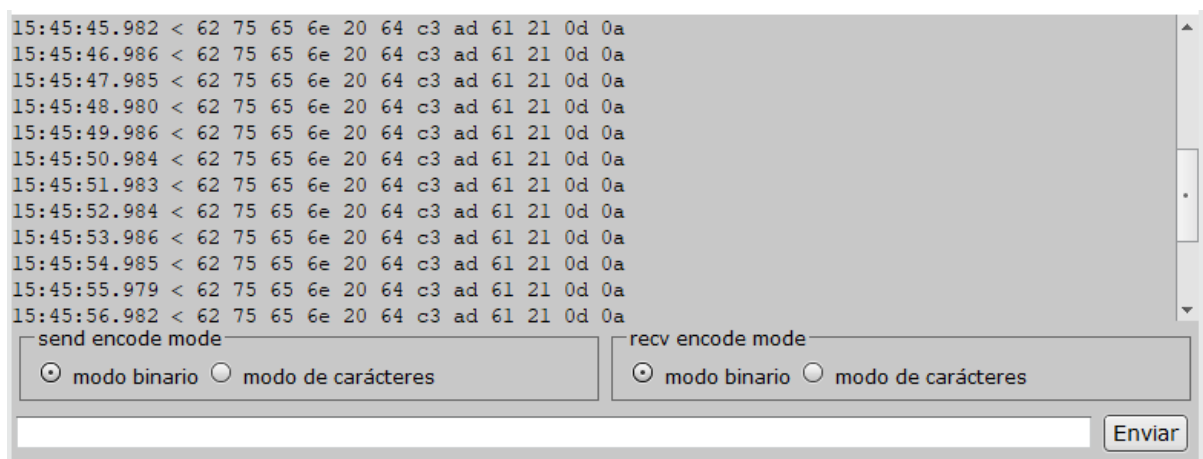




Fig. 10

Una vez seleccionado el “modo de caracteres”, deberíamos poder leer los mensajes correctamente, como se ve a continuación:

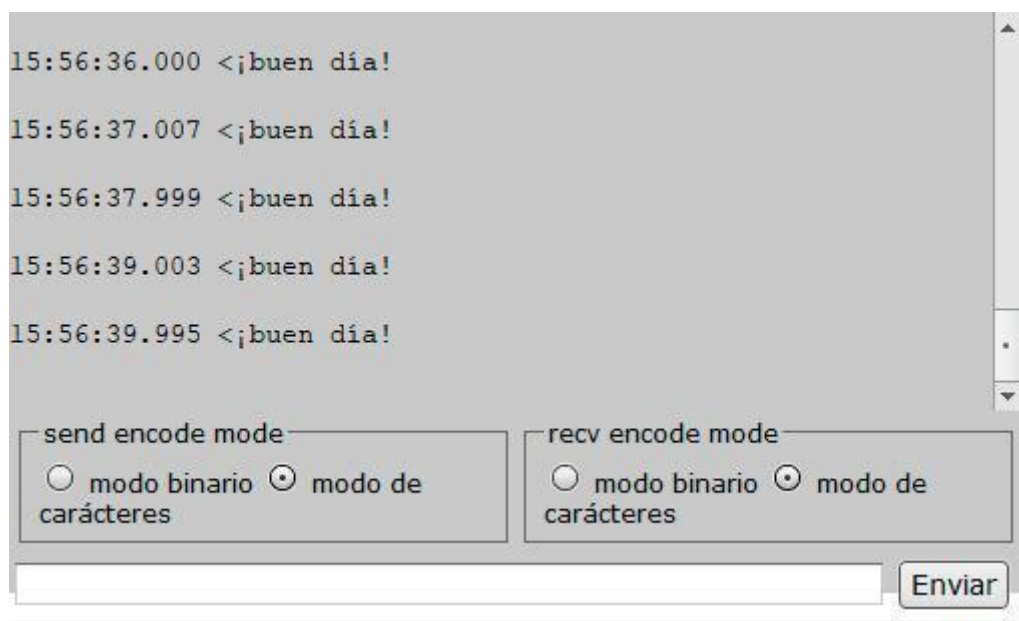


Fig. 11

### Paso 3 - Leer el higrómetro por el puerto serie

Ahora que hemos visto cómo comunicarnos con la computadora por medio del puerto serie, podemos enviar otros mensajes. En particular, vamos a transmitir a la computadora los valores que leemos en la entrada analógica A0, en donde tenemos conectado el higrómetro.

Modificamos nuestro programa para que en lugar de escribir “¡buen día!” escriba el valor leído en la entrada analógica A0. Nuestro programa entonces queda:

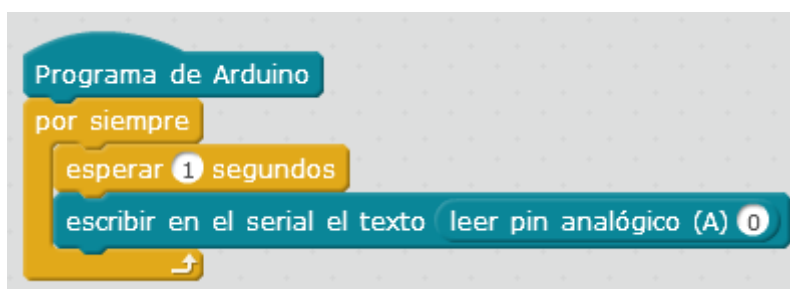


Fig. 12

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>

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

void setup(){
    Serial.begin(115200);
    pinMode(A0+0,INPUT);
}

void loop(){
    _delay(1);
    Serial.println(analogRead(A0+0));
    _loop();
}

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

void _loop(){
}

```

Fragmento de código 4

Cada vez que subimos el programa a la placa tenemos que reconectarnos al puerto serie para recibir los mensajes. Recuerden que se utiliza el mismo puerto para la comunicación y para subir los programas. Por eso, antes de subir el programa, mBlock3 nos desconecta del puerto serie. Para volver a conectarnos seleccionamos el puerto serie desde el menú:

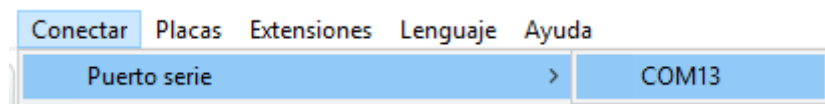


Fig. 13

Ahora deberíamos recibir en la computadora valores entre 0 y 1023 según la humedad que detecte el higrómetro.

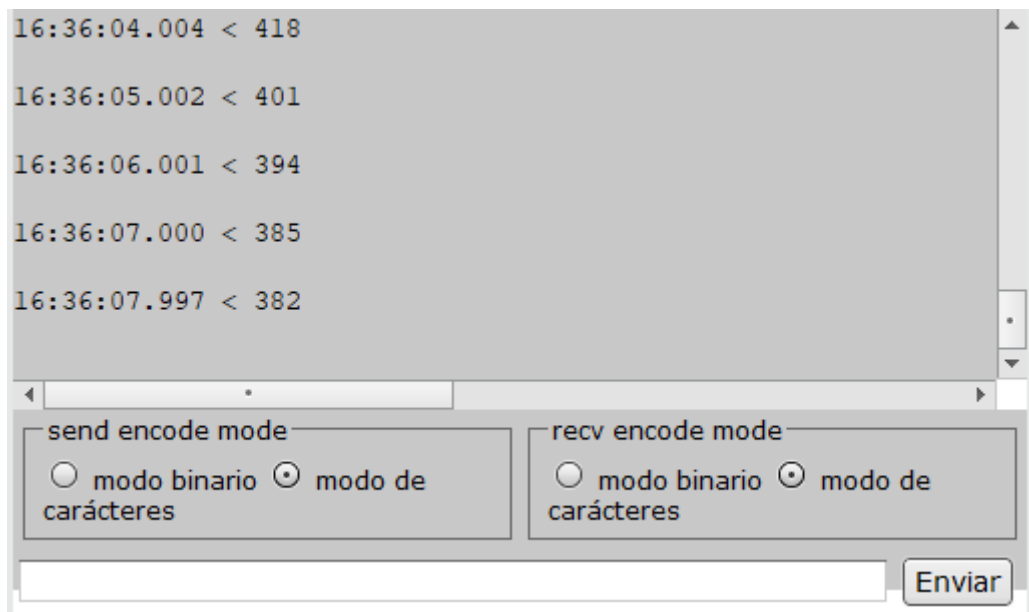


Fig. 14

#### Paso 4 - Determinar los valores de humedad para activar la válvula

El uso del higrómetro nos va a permitir determinar si debemos abrir la válvula, cuando el suelo esté demasiado seco; o cerrarla, cuando esté suficientemente húmedo.

En primer lugar debemos determinar ante qué valores consideraremos que el suelo se encuentra “demasiado seco” y ante cuáles “suficientemente húmedo”. Ahora que ya podemos ver qué valores arroja el higrómetro, tenemos la posibilidad de tomar muestras del suelo con diferentes grados de humedad en unos recipientes, colocarles el higrómetro e ir tomando nota de los valores que vemos por medio del puerto serie. No debemos olvidar limpiar y secar bien el higrómetro entre una muestra y la siguiente.

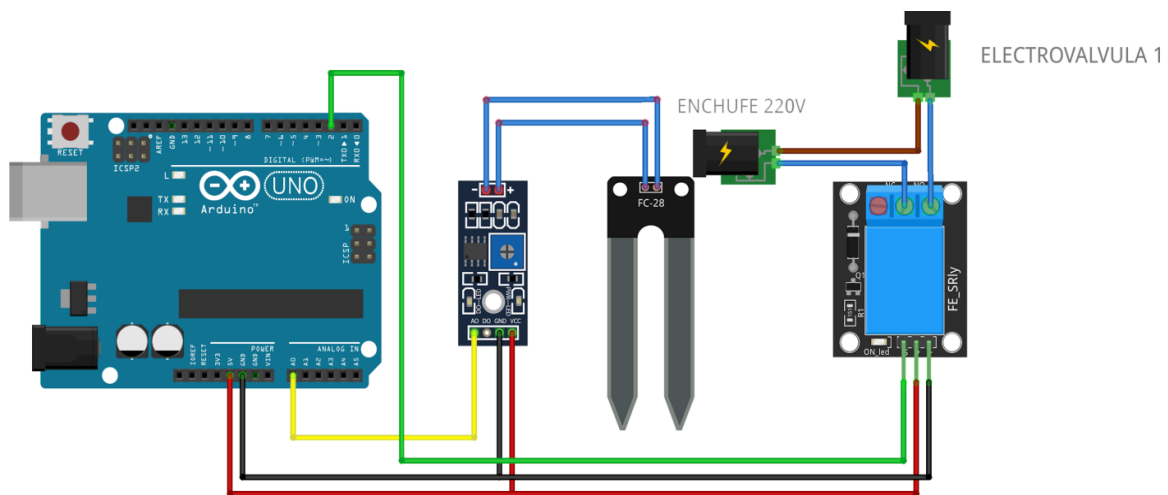
Supongamos que luego de varias muestras y promedios, determinamos que los valores de la entrada analógica A0 indican lo siguiente:

- mayor a 750 => demasiado seco
- menor a 380 => suficientemente húmedo

Con estos valores determinaremos las condiciones que deben cumplirse para que nuestro sistema abra o cierre la válvula.

#### Paso 5 - Programar la apertura y cierre de la válvula en base a la humedad del suelo

Conectamos el higrómetro y la válvula como muestra el siguiente esquema:



Esquema 3

**¡Atención! Para construir este dispositivo trabajaremos con una tensión de 220V. Si se está trabajando con protoboard, se recomienda no incluir en el mismo las conexiones a relé y 220V.**

En base a las mediciones efectuadas en el paso anterior, nuestro programa debería ser similar al siguiente modelo:



Fig. 15

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>

double angle_rad = PI/180.0;
```

```

double angle_deg = 180.0/PI;

void setup(){
  Serial.begin(115200);
  pinMode(A0+0,INPUT);
  pinMode(2,OUTPUT);
}

void loop(){
  _delay(1);
  Serial.println(analogRead(A0+0));
  if((analogRead(A0+0)) > (750)){
    digitalWrite(2,1);
  }
  if((analogRead(A0+0)) < (380)){
    digitalWrite(2,0);
  }
  _loop();
}

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

void _loop(){
}

```

Fragmento de código 5

Cada un segundo nuestro programa nos informará por medio del puerto serie el valor leído por el higrómetro, y abrirá o cerrará la válvula de acuerdo al estado de humedad de la tierra.

### Nivel Avanzado

En última instancia, estudiantes y docentes se propusieron generar un sistema que les permitiera monitorear remotamente, desde un dispositivo móvil, el nivel de humedad del suelo y el estado de la válvula.

**En esta tercera instancia se programará el envío de datos obtenidos a un dispositivo móvil a través de internet (IoT).**

## 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.

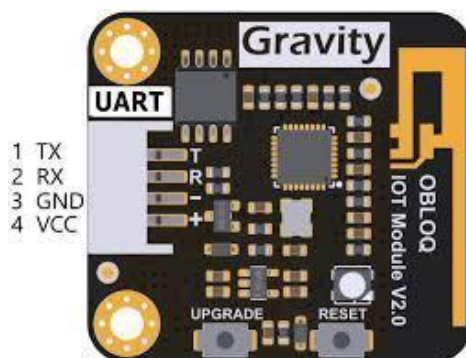


Fig. 16

## 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](https://io.adafruit.com).

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

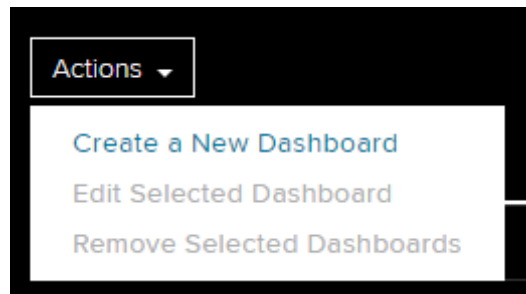


Fig. 17

Creamos un nombre y una descripción. Y al finalizar presionamos el botón “Create”.

A light-themed modal window titled 'Create a new Dashboard' with a blue close button in the top right. It contains two text input fields: 'Name' with the value 'Riego Automatizado' and 'Description' with the value 'Diseño, construcción y programación de un sistema de riego automatizado.' At the bottom right are two buttons: a grey 'Cancel' button and a blue 'Create' button.

Fig. 18


Hacemos click en el nuevo panel creado y veremos una pantalla vacía. Podemos comenzar a agregar bloques haciendo click en 



Fig. 19

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

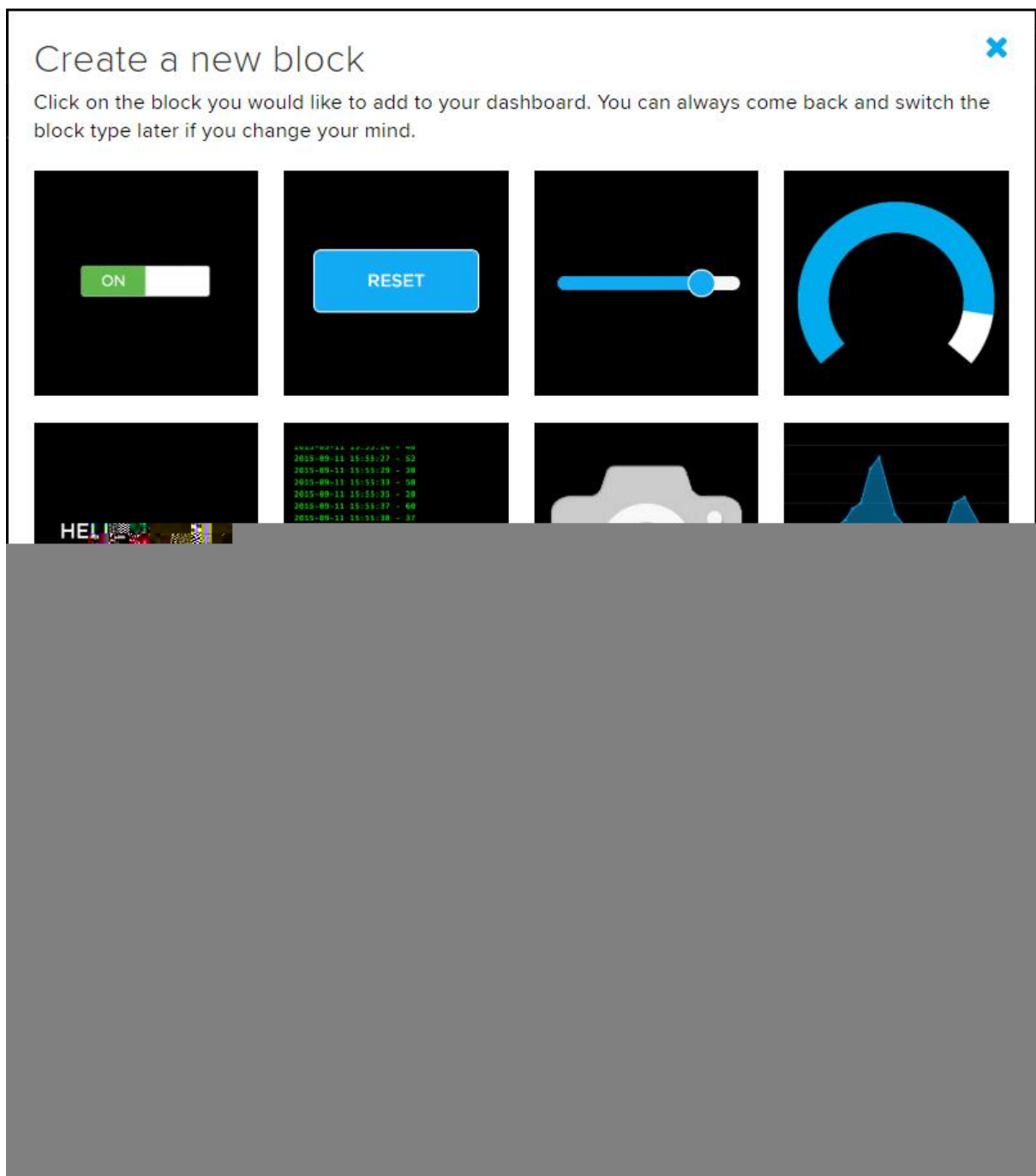


Fig. 20

Para nuestro sistema de riego, podríamos ubicar un *Gauge*, para mostrar el nivel de humedad, y un *Indicator* (indicador), para mostrar si la válvula está o no abierta.





**Gauge** Fig. 21



**Indicator**  
Fig. 21

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* “humedad” y le damos al botón “Create”. En él publicaremos, desde nuestro dispositivo, los valores leídos en la entrada analógica donde tenemos conectado el higrómetro.

Choose feed

**Gauge:** A gauge is a read only block type that shows a fixed range of values.

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	🔒	10 days

Fig. 22

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

Choose feed

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 checked="" type="checkbox"/> humedad	🔒	less than a minute
<input type="checkbox"/> Welcome Feed	🔒	10 days

Fig. 23

Ajustamos los parámetros del bloque como se ve en la imagen:

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)

Humedad del Suelo

Gauge Min Value

0

Gauge Max Value

1023

Gauge Width

25px

Gauge Label

Low Warning Value

380

Optional. If no low warning value is given, the gauge will only change color when the value is out of bounds.

High Warning Value

750

Optional. If no high warning value is given, the gauge will only change color when the value is out of bounds.

Block Preview

Humedad del Suelo

380

750

1023

0

450

Gauge

A gauge is a read only block type that shows a fixed range of values.

Test Value

450

Published Value

0 bytes

Previous step

Create block

Fig. 24

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

Si se quiere, podemos modificar el tamaño y ubicación de los bloques haciendo click en "la rueda de configuración".

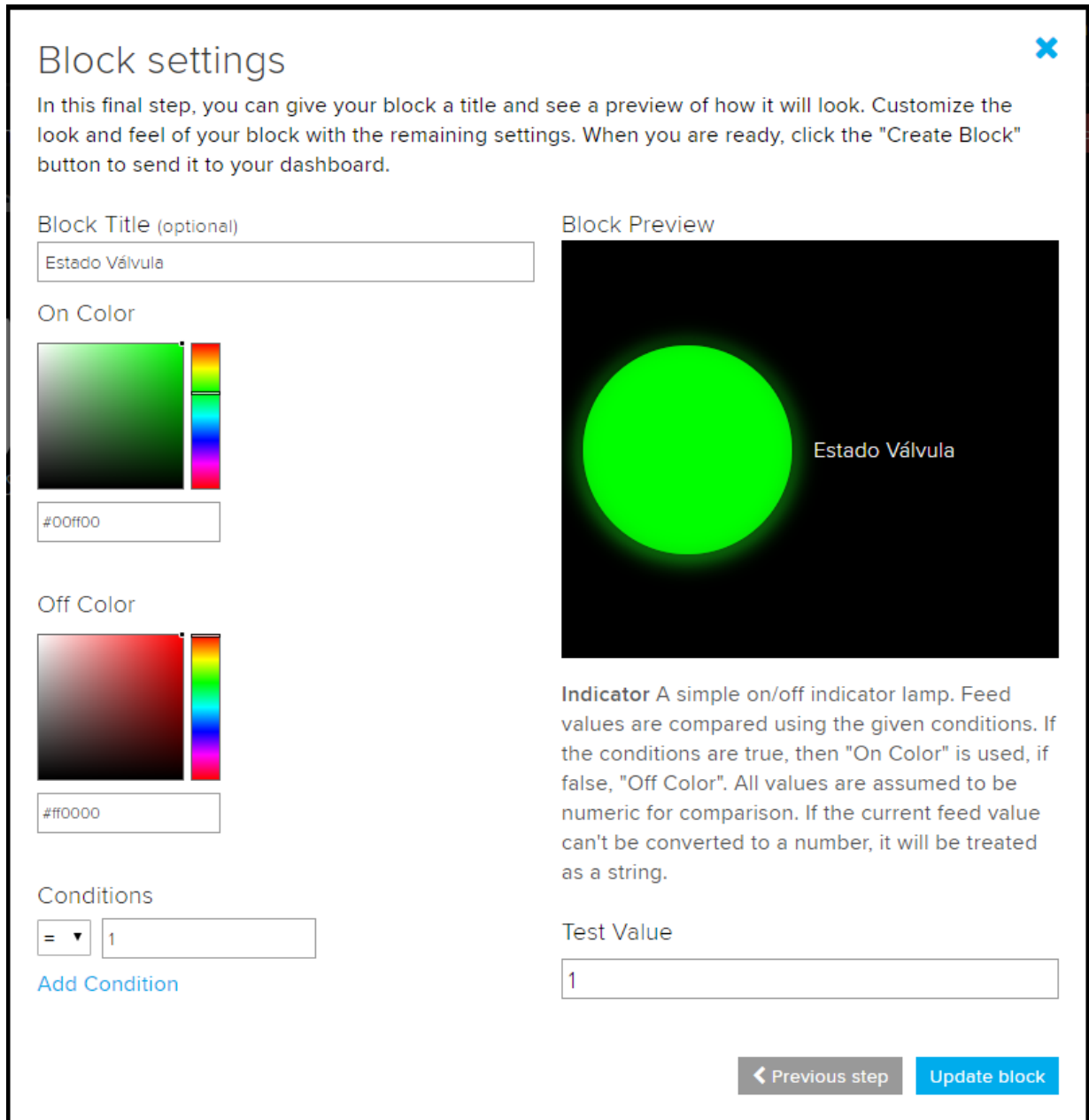
Fig. 25

26



Repetimos este procedimiento para el *feed* “válvula” (sin tilde) que indicará si la válvula está abierta (cuando el *feed* valga 1) o cerrada (cuando el *feed* valga 0).

Ajustamos los parámetros del bloque como se ve en la imagen:



**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)  
Estado Válvula

On Color  
#00ff00

Off Color  
#ff0000

Conditions  
= 1  
[Add Condition](#)

Block Preview

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](#) [Update block](#)

Fig. 25

Luego, hacemos click en “Create block” (crear bloque) para completar la operación.

Si se quiere, podemos modificar el tamaño y la ubicación de los bloques haciendo click en la “rueda de configuración”.

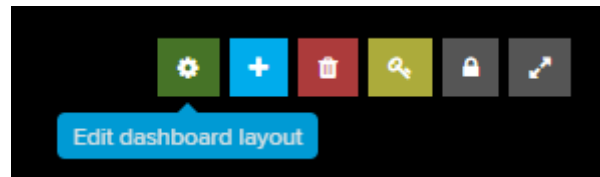


Fig. 26

Se debería ver algo similar a lo siguiente:

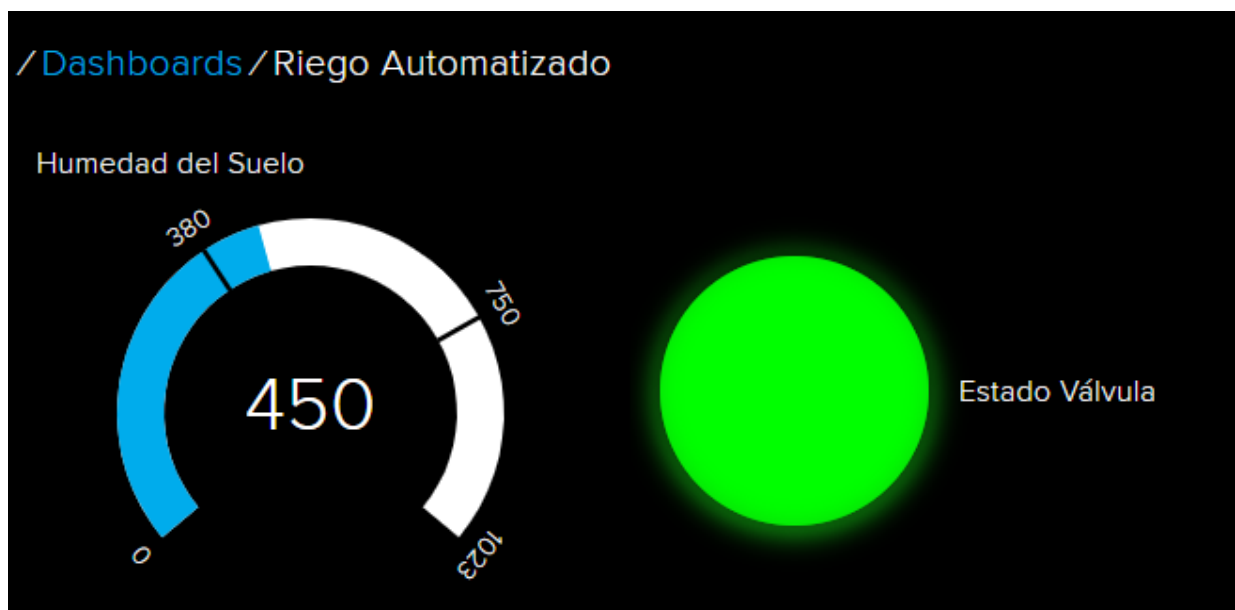
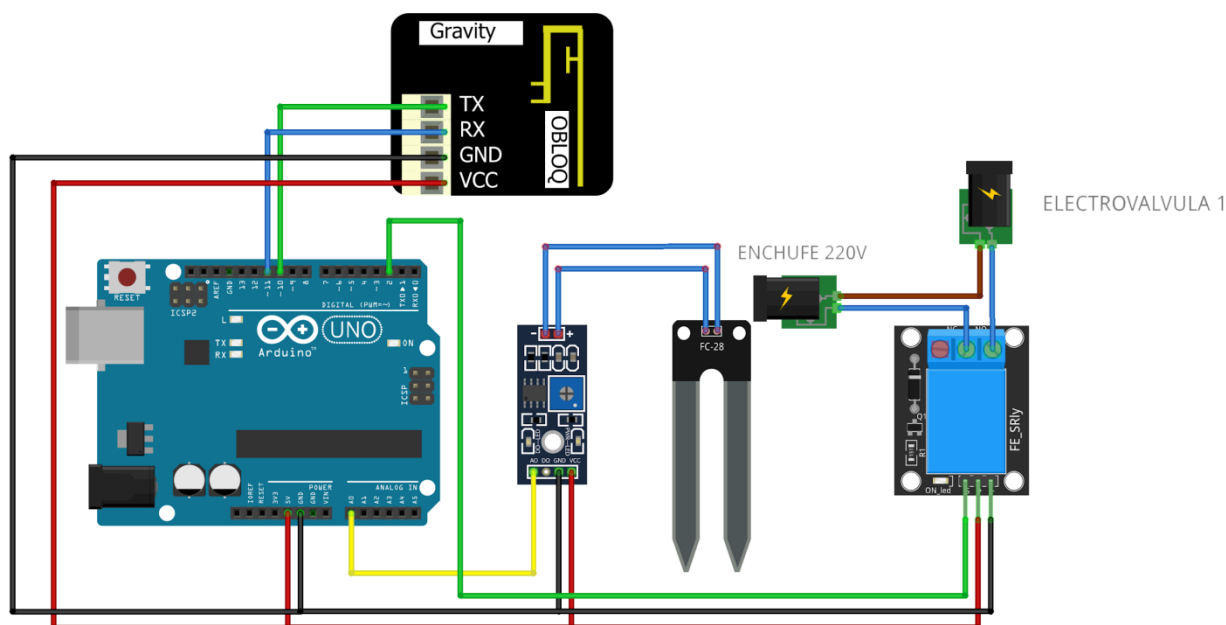


Fig. 27

Una vez realizado el panel, publicaremos en nuestros feeds el estado de la válvula (abierto o cerrado) y el nivel de humedad del suelo para poder monitorearlos de manera remota.

### Paso 3 - Agregar módulo OBLOQ

Conectamos el módulo OBLOQ como se muestra en el esquema siguiente:



Esquema 4

### Paso 4 - Arduino IDE

La programación por bloques tiene sus ventajas desde un punto de vista didáctico, pero tiene sus límites. Cuando el programa que queremos realizar comienza a ser más complejo, podemos encontrarnos con el hecho de que ciertas operaciones no pueden resolverse utilizando bloques o que, hacerlo con este método, resulta en algo mucho más engorroso y difícil de leer 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](http://www.enfoco.net.ar/sd)

Veremos que se nos presenta la siguiente interfaz:

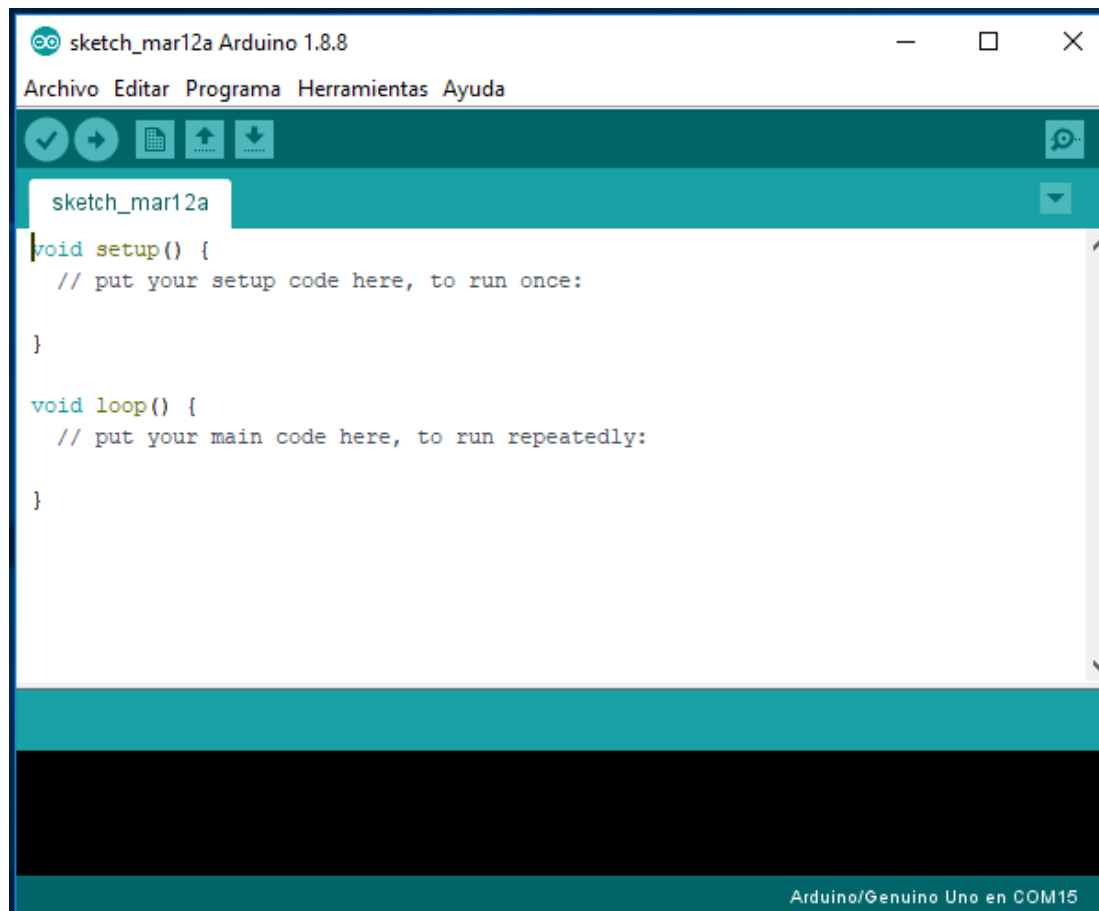


Fig. 28

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 esté seleccionada la placa “Arduino/Genuino Uno” y el puerto serie al que está conectada la placa

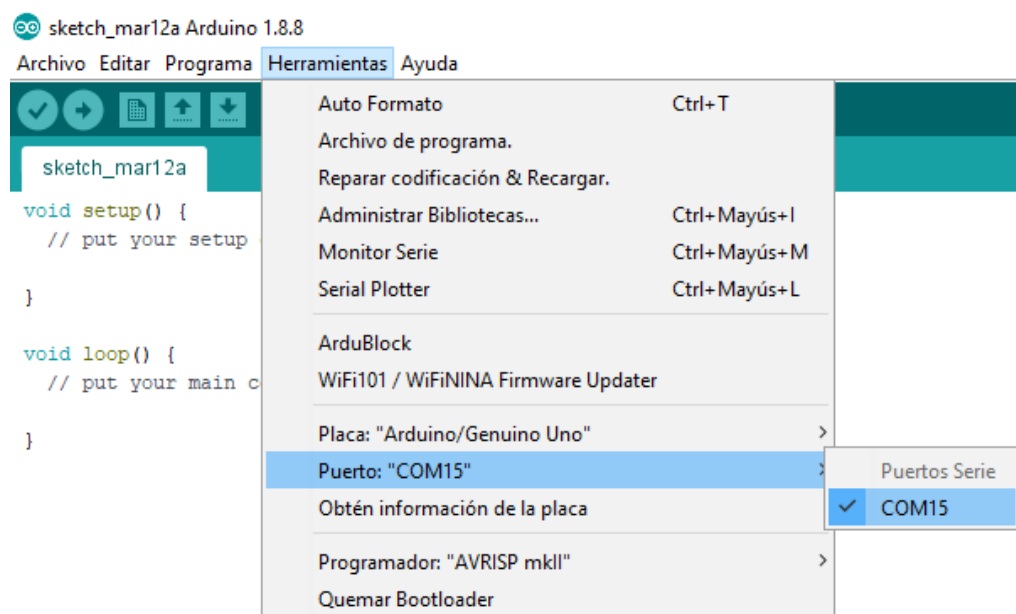


Fig. 29

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 5 - Programar sin código bloqueante

Antes de comenzar a utilizar IoT debemos hacer una aclaración con respecto a la función `_delay()` que figura en el código que usamos hasta ahora. Esta función brinda un tiempo de espera al sistema que puede utilizarse con varios fines. Suele utilizarse bastante en las primeras aproximaciones a la programación ya que su comportamiento resulta fácil de comprender y su programación no requiere más que una línea de código. Sin embargo, esta función tiene una complicación, dado que genera un “código

bloqueante". Esto significa que, cuando el programa entra en esa función, se detiene todo el procesamiento hasta que se cumpla el tiempo indicado. En otras palabras, cuando el programa entra al *delay* queda "colgado" por el período de tiempo establecido.

Al utilizar IoT, es conflictivo utilizar código "bloqueante" ya que, al detenerse el procesamiento, se impide también que el sistema realice otras operaciones que funcionan en simultáneo. Por ejemplo, las tareas de publicación y el mantenimiento constante de la conexión a internet.

Para evitar estos problemas, se puede utilizar una alternativa de código "no bloqueante", como la función `millis()`. Esta función arroja un valor sobre un conteo de tiempo, que se realiza desde el momento en que se inicia el sistema. Es decir, funciona como un cronómetro (en milisegundos) que, cada vez que es consultada desde el código, "devuelve" el valor en el que se encuentra. De esta manera podemos pedirle al sistema que informe cuánto tiempo transcurrió desde el inicio de las operaciones para dar indicaciones temporales sobre una tarea, sin detener todas las demás.

A continuación se presenta un ejemplo de cómo se puede programar la intermitencia de un LED que se prenda y apague cada un segundo (expresado en 1000 milisegundos) sin utilizar código bloqueante. Lo haremos utilizando la función *millis* para consultar cuánto tiempo pasó.

```
int estado = LOW;
// Se declara "millisAnterior" con valor inicial igual a cero.
long millisAnterior = 0;

void setup() {
  // Inicializa el pin digital 13 como una salida.
  pinMode(13, OUTPUT);
}

void loop() {

  long millisActual = millis();

  if (millisActual - millisAnterior >= 1000) {
    // Conmuta el estado del led.
    if (estado == LOW) {
      estado = HIGH;
    } else {
      estado = LOW;
    }

    // Setea el estado del led.
  }
```

```
digitalWrite(13, estado);

// Guarda la última vez que conmutamos el led.
millisAnterior = millisActual;
}

// Y en este punto nuestro procesador queda libre
// para realizar otras tareas.

}
```

#### Fragmento de código 6

En el ejemplo se puede observar que para tomar el valor de *millis* se define un valor inicial, al que llamamos “millisAnterior”, que es igual a cero. Luego, para consultar el tiempo transcurrido desde el inicio del sistema, se calcula la diferencia entre el valor de “millisActual” y “millisAnterior”. En el caso de nuestro ejemplo, como queremos generar una intermitencia de un segundo, necesitamos evaluar si esta diferencia es mayor o igual a 1000. En caso de que haya transcurrido más de un segundo, el sistema modifica el estado de la luz. Si ha transcurrido menos tiempo, el estado se mantiene estable.

En última instancia, se establece que, si esta diferencia es mayor o igual a 1000, se le asigne a “millisAnterior” el valor de “millisActual”. De esta manera, la diferencia entre “millisActual” y “millisAnterior” vuelve a ser cero hasta que vuelva a transcurrir otro segundo.

## Paso 6 - Programación IoT

Utilizaremos la librería ObloqAdafruit para informar a Adafruit el valor de humedad del suelo y el estado de la válvula. Podremos monitorear estos datos 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.

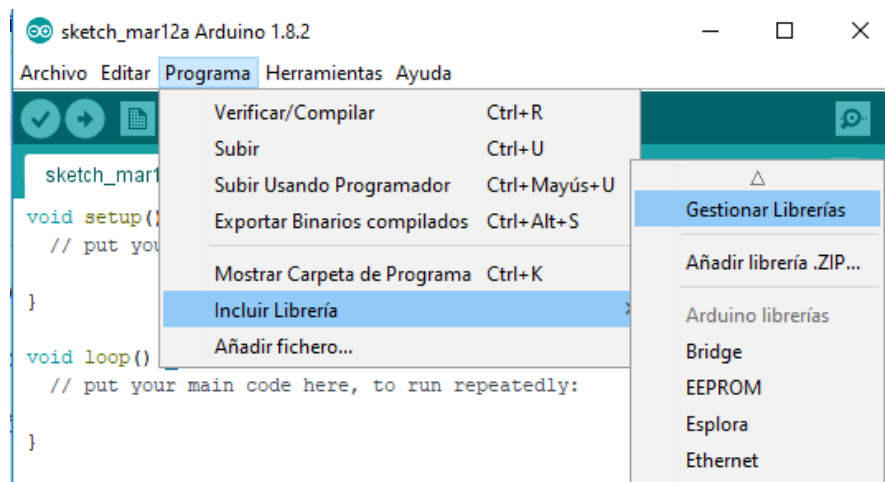


Fig. 30

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

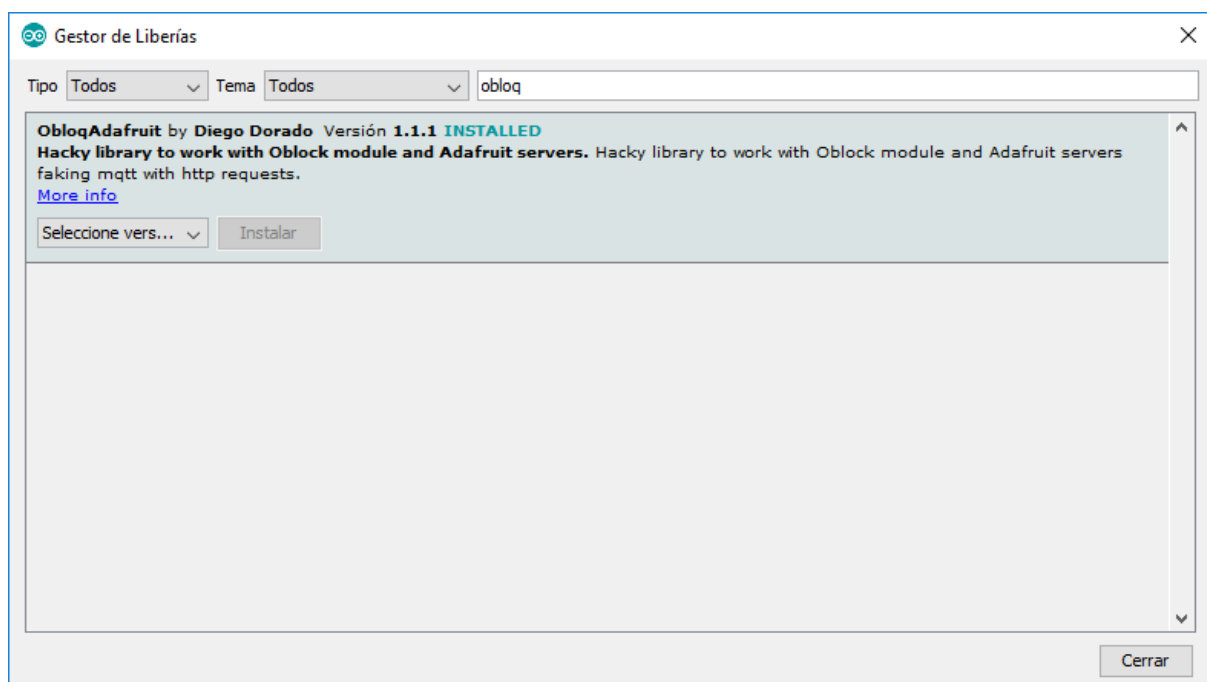


Fig. 31

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. Pare ello maximizamos la pantalla de Adafruit y hacemos click en el ícono de la “llave”.

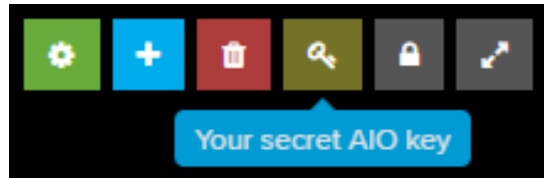


Fig. 32

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

Active Key

REGENERATE AIO KEY

[Hide Code Samples](#)

**Arduino**

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

Fig. 33

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(6,7)` 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(6,7);
ObloqAdafruit olq(&softSerial,WIFI_SSID,WIFI_PASSWORD,IO_USERNAME,IO_KEY);
```

El *setup* debe incluir la línea de inicialización del softwareSerial:

```
void setup()
{
    softSerial.begin(9600);
}
```

Se debe agregar también la función de actualización o “update”: `olq.update()`. Por esto es importante que nuestro código no sea bloqueante.

```
void loop()
{
    olq.update();
    // ..
    // ..
}
```

Para publicar un *feed*, utilizaremos la función `publish` del objeto `olq` :

```
olq.publish("valvula", 1); // informar que la válvula está abierta
```

Finalmente, nuestro programa de semáforo con IoT queda como sigue:

```
#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(10,11);
ObloqAdafruit olq(&softSerial,WIFI_SSID,WIFI_PASSWORD,IO_USERNAME,IO_KEY);

long millisAnterior = 0;

void setup(){
  pinMode(A0,INPUT);
  pinMode(2,OUTPUT);

  // Inicializamos la comunicación con el módulo OBloq.
  softSerial.begin(9600);
}

void loop()
{
  olq.update();

  // No usar delay!, chequear el tiempo transcurrido con millis().
  if(millis() - millisAnterior > 1000)
  {
    int humedad = analogRead(A0);

    if( humedad > 750){
      digitalWrite(2,1);
      olq.publish("valvula", 1);
    }
    if( humedad < 380){
      digitalWrite(2,0);
      olq.publish("valvula", 0);
    }

    olq.publish("humedad", humedad);
    millisAnterior = millis();
  }
}

```

Fragmento de código 7

## Cierre

Una vez finalizado este proyecto, si se quiere continuar, es posible extenderlo. Una opción sugerida es:

- Complementar con el proyecto “Invernadero inteligente”, para agregar sensores de humedad de ambiente y de temperatura.

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.).

**Te recomendamos pasar por el repositorio de saberes Digitales para obtener más materiales y otros ejemplos: [www.enfoco.net.ar/sd](http://www.enfoco.net.ar/sd)**



# 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 (entorno de desarrollo integrado) para generar el código, compilarlo y quemarlo en la placa. Existen múltiples IDE compatibles con las placas Arduino.

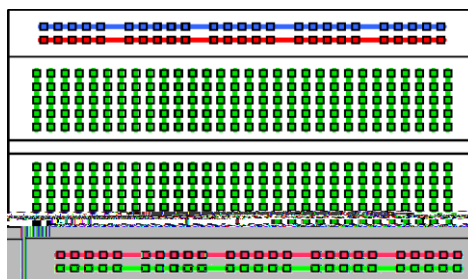
El microcontrolador posee entradas analógicas y digitales así como salidas digitales, PWM y servo. Las entradas y salidas digitales son las que permiten leer o escribir estados del tipo binarios. Pueden adoptar la forma de 0 ó 1, alto o bajo, verdadero o falso. Para prender y apagar los LED del semáforo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



**Higrómetro de suelo:** sensor que registra la humedad del suelo mediante la medición de la conductividad eléctrica. El mismo se compone de una lanza que se clava en la tierra y tiene contactos que cierran un circuito eléctrico en relación a la humedad de la misma. Este tipo de sensor no tiene la sensibilidad suficiente como para realizar una medición precisa de la humedad pero sí se puede usar para diferenciar un suelo seco de uno húmedo. El higrómetro de suelo tiene 3 pines de conexión, dos se utilizan para la alimentación eléctrica

(VCC y GND) y el tercer pin es la salida de información. La salida del mismo es analógica, entregando un nivel de tensión en relación a la humedad registrada. Se debe conectar a una entrada analógica del Arduino para poder leer la información proporcionada.

**Electroválvula:** Se trata de un actuador que permite interrumpir o no el paso de un fluido a través del mismo. Existen versiones que trabajan con 12 V así como también los que trabajan directamente con 220V, permitiendo trabajar en un sistema de mayor nivel de caudal. El accionamiento del mismo se realiza directamente a través de la alimentación de la misma, por lo que tiene únicamente 2 pines para la conexión. Los estados posibles son activado o no, por lo que se puede controlar mediante una salida digital y un relé.

**Relé:** Dispositivo electromagnético que funciona como un interruptor controlado por un circuito eléctrico. Por medio de un electroimán se acciona uno o varios contactos que permiten abrir o cerrar otros circuitos eléctricos independientes. Se utiliza comúnmente para aislar eléctricamente dos circuitos así como también permitir controlar con bajo voltaje a dispositivos que utilizan mayor voltaje. Arduino trabaja con señales de 5V que mediante un relé permiten controlar la activación o desactivación de dispositivos de 220V.

## Internet de las cosas

**Panel de Control Adafruit:** Los sistemas IoT trabajan apoyándose en un servidor que se encarga de gestionar la información que reportan los diversos sensores así como responder a las consultas de los dispositivos que buscan acceder a esta 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, 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!***