

Práctica 1. Introducción a Arduino.

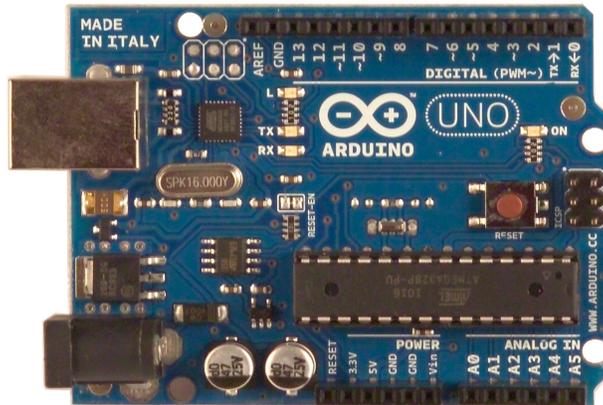
Objetivos:

1. Familiarizar al alumno con la tarjeta de desarrollo Arduino y su entorno de desarrollo (IDE).
2. El alumno aprenderá la forma de conectar sensores y actuadores simples a la tarjeta Arduino.
3. El alumno aprenderá las técnicas de programación básicas que le permitan resolver problemas prácticos.

Desarrollo

1. Descripción de la tarjeta

La tarjeta Arduino es una plataforma de desarrollo basada en un microcontrolador, en el caso del modelo UNO, el microcontrolador ATmega328. Arduino es un proyecto de tipo *open-source*, diseñado para simplificar el proceso de desarrollo de sistemas basados en microcontroladores, ofreciendo con esto muchas ventajas a estudiantes, profesores, y en general entusiastas del movimiento *Maker*.



Tarjeta de Desarrollo Arduino UNO

La plataforma Arduino es una de las herramientas de desarrollo más utilizadas debido a la amplia oferta de productos compatibles (*shields*), bibliotecas de software disponibles, su facilidad de programación, versatilidad, y la gran cantidad de usuarios que comparten su conocimiento.

2. Instalación del entorno de desarrollo integrado (IDE)

El entorno de desarrollo integrado de Arduino es el programa que permite el desarrollo de software y su posterior descarga en la tarjeta. Este programa está disponible para los sistemas operativos Windows, Mac OS y Linux.

Descargue la versión adecuada para el sistema operativo y arquitectura (32 o 64 bits) de su PC. Podrá encontrar las ligas de descarga en la siguiente dirección:

<http://arduino.cc/en/Main/Software>

Para el sistema operativo Windows, descargue el instalador, ejecútelo y siga las instrucciones en pantalla. Una vez terminado este proceso, será necesaria la instalación del driver de la tarjeta Arduino.

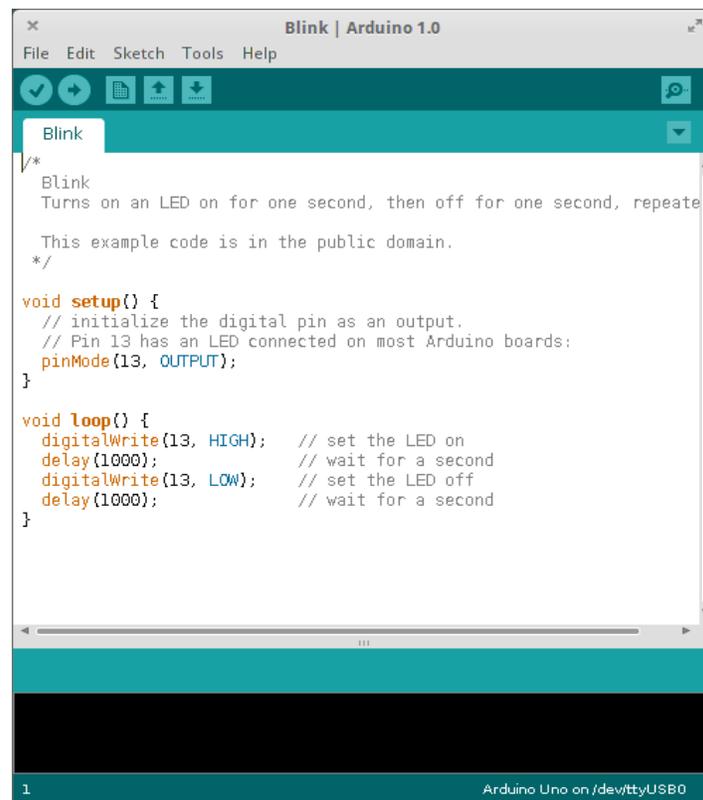
Abra la ventana del Administrador de Dispositivos y localice el puerto *Arduino UNO* (*COMxx*). Haga click con el botón derecho del ratón y seleccione la opción de actualizar el controlador. Posteriormente elija la opción de buscar archivo del controlador manualmente y seleccione el archivo *arduino.inf*.

Para el sistema operativo Linux, puede optar por descargar el paquete correspondiente de la liga anterior o bien ejecutar el comando siguiente en una terminal:

```
sudo apt-get install arduino arduino-core
```

Después de ejecutar el comando, el entorno de desarrollo quedará instalado dentro del sistema operativo, lo que no ocurre al descargar el paquete desde el sitio web de Arduino. En Linux no es necesaria la instalación de drivers adicionales.

Una vez conectada la tarjeta Arduino a su PC a través del cable USB, ejecute el entorno de desarrollo (IDE).

The image shows a screenshot of the Arduino IDE interface. The window title is "Blink | Arduino 1.0". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for a checkmark, a play button, a document with a plus sign, an upload button, and a download button. The main text area contains the following code:

```
/*  
 * Blink  
 * Turns on an LED on for one second, then off for one second, repeats  
 *  
 * This example code is in the public domain.  
 */  
  
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH); // set the LED on  
  delay(1000);           // wait for a second  
  digitalWrite(13, LOW); // set the LED off  
  delay(1000);          // wait for a second  
}
```

At the bottom of the window, there is a status bar that reads "1" on the left and "Arduino Uno on /dev/ttyUSB0" on the right.

IDE de Arduino ejecutándose en un sistema Linux

3. Descarga y análisis de programas de ejemplo

Dentro del entorno de desarrollo abra el ejemplo *Blink*. Seleccione el modelo de su tarjeta y el puerto serie asociado a ella. Verifique el código y descárguelo en la tarjeta Arduino. Realice lo siguiente:

1. Describa el funcionamiento práctico del ejemplo.
2. Identifique la estructura básica de un programa escrito para el IDE de Arduino (*sketch*).
3. Describa el funcionamiento del programa. Ayúdese con los comentarios en el código.
4. Modifique el programa para utilizar un pin diferente al del ejemplo.
5. Conecte adecuadamente un *LED* al pin seleccionado (ver apéndice). A continuación verifique y descargue el nuevo programa a la tarjeta Arduino.

Dentro del entorno de desarrollo abra el ejemplo *DigitalReadSerial*. Realice lo siguiente:

1. Conecte adecuadamente el sensor magnético proporcionado, al pin indicado en el programa (ver apéndice).
2. Verifique el código y descárguelo en la tarjeta Arduino.
3. Abra el monitor serial, acerque y aleje el imán. Describa el funcionamiento práctico del ejemplo.
4. Analice el código y describa el funcionamiento del programa.

Dentro del entorno de desarrollo abra el ejemplo *switchCase2*. Realice lo siguiente:

1. Analice el código y describa el funcionamiento del programa.
2. Modifique el programa para que únicamente reconozca dos caracteres diferentes.
3. Conecte adecuadamente dos *LEDs* a los pines definidos en su programa. A continuación verifique el código y descárguelo en la tarjeta Arduino.
4. Abra el monitor serial. Ingrese, uno a la vez (seguidos de un *Enter*), los caracteres reconocibles por su programa. Describa el funcionamiento práctico del ejemplo.

Dentro del entorno de desarrollo abra el ejemplo *SerialEvent*. Realice lo siguiente:

1. Verifique el código y descárguelo en la tarjeta Arduino.
2. Abra el monitor serial. Ingrese una palabra y a continuación presione *Enter*. Describa el funcionamiento práctico del ejemplo.
3. Analice el código y describa el funcionamiento del programa. Ayúdese con los comentarios en el código.

4. Desarrollo de software

Realice lo siguiente:

1. Escriba un programa para Arduino que controle el encendido y apagado de un *LED* mediante el sensor magnético, ambos conectados a la tarjeta Arduino. Cuando el imán se encuentre cerca del sensor el *LED* deberá permanecer encendido y viceversa.
2. Escriba un programa para Arduino que además de controlar el encendido y apagado de

un *LED* mediante el sensor magnético, envíe el valor digital de este último por el puerto serie. En caso de tener el imán cerca del sensor, se deberá visualizar en el monitor serial un 1, en caso contrario un 0.

3. Sin desconectar el *LED* indicador, escriba un programa para Arduino que identifique el comando ***magnet*** recibido a través del puerto serie y en seguida devuelva por el mismo puerto el valor digital del sensor magnético (0 o 1).
4. Escriba un programa para Arduino que identifique el comando ***led on/off*** recibido a través del puerto serie y en seguida encienda o apague un *LED*. Al detectar el comando ***led on***, el *LED* deberá encenderse; al detectar ***led off***, el *LED* deberá apagarse.

Apéndice. Conexión de interruptores y LEDs a la tarjeta Arduino.

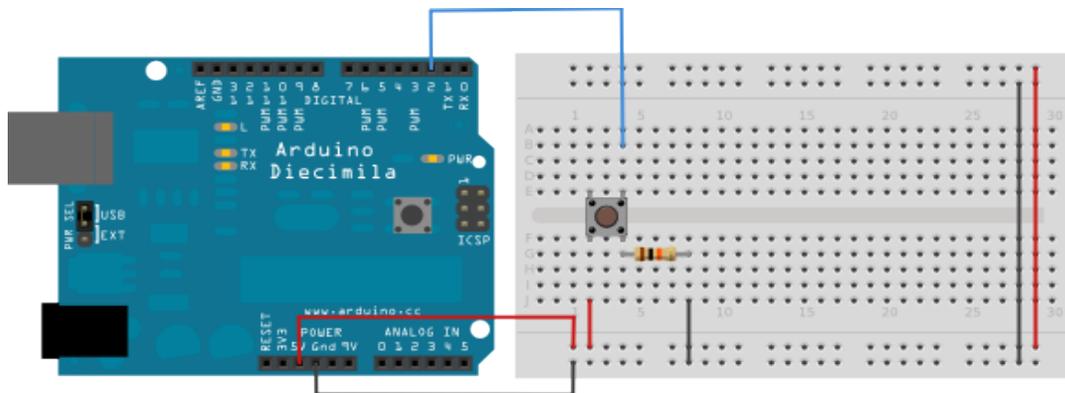
Interruptores

La conexión de cualquier interruptor a la tarjeta Arduino debe hacerse utilizando un resistor, configuración conocida como *pull-up/down*. Con esto se asegura que en la terminal del microcontrolador exista siempre un voltaje bien definido.

Se conoce como *pull-up* al circuito que conecta la terminal del microcontrolador, a través de un resistor, al voltaje correspondiente a un 1 lógico, comúnmente 5 o 3.3 [V]. Para este caso, el interruptor deberá situarse entre la terminal del microcontrolador y tierra.

En contraste, *pull-down* se refiere a la conexión de la terminal del microcontrolador, a través de un resistor, al voltaje correspondiente a un 0 lógico o tierra (GND), es decir 0 [V]. Para este caso, el interruptor deberá situarse entre la terminal del microcontrolador y el voltaje de 1 lógico.

El valor del resistor no es crítico. Un resistor de 10 [k Ω] es de un valor adecuado.



Conexión de un push button a la tarjeta Arduino en modo pull-down

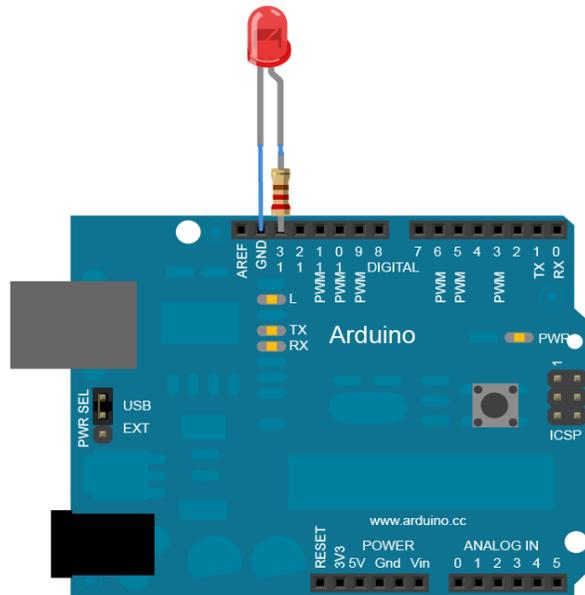
Diodos emisores de luz (LED)

La forma correcta de conectar un *LED* a un microcontrolador es a través de un resistor que limite la corriente que pasará a través del *LED*. El valor de este resistor depende de los voltajes que maneje el microcontrolador y la corriente necesaria para encender el *LED*.

Particularmente, para conectar un *LED* convencional a la tarjeta Arduino (5 [V]), se recomienda utilizar valores de resistencia de 220 o 330 [Ω].

Un aspecto importante a tomar en cuenta es que los *LEDs*, como cualquier diodo sólo pueden conducir corriente en un sentido. La terminal más larga de un *LED* corresponde a su ánodo, y debe conectarse a un voltaje superior al de la terminal corta (cátodo) para poder encenderse.

En términos prácticos, la terminal corta del *LED* debe conectarse a tierra (0 [V]), mientras la terminal larga debe conectarse, a través del resistor, a la terminal de la tarjeta Arduino que deseamos que controle el encendido y apagado del *LED*; si conectamos sus terminales en sentido inverso, el *LED* simplemente no se encenderá nunca.



Conexión de un LED a la tarjeta Arduino

Práctica 2. Comunicación utilizando el lenguaje Python.

Objetivo:

- El alumno utilizará el lenguaje de programación Python para recibir y enviar comandos a la tarjeta de desarrollo Arduino.

Introducción

Python es un lenguaje de programación creado en 1991 cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Esto implica que la sintaxis del lenguaje es simple y cercana al lenguaje natural, en lugar de utilizar símbolos para representar operadores Python utiliza palabras tales como “OR”, “AND”, “NOT”; la mayor ventaja de utilizar Python es el hecho de que es muy sencillo llegar a dominarlo e incluso se suele decir que es más fácil aprenderlo para una persona que nunca ha programado que para una que ya conoce otros lenguajes de programación.

Desarrollo

Lectura e impresión:

1. Localice y abra el editor de texto **gedit**. Escriba las siguientes líneas de código.

```
# -*- coding: latin-1 -*-  
  
# lee desde el teclado hasta encontrar un salto de linea  
(ENTER)  
# y lo almacena en s:  
s = raw_input("->")  
  
# imprime en pantalla el valor de s:  
print "Escribiste: " + s
```

Guarde el archivo con el nombre **std_in_out.py** en la Carpeta Personal del usuario.

2. Analice el código y describa el funcionamiento del programa.
3. Para comprobar el funcionamiento del código: abra una terminal de comandos del sistema. Teclee:

python std_in_out.py

Ingrese texto desde el teclado y presione ENTER para terminar.

Escritura serial:

1. Cree un nuevo archivo y en él escriba las siguientes líneas de código.

```
# -*- coding: latin-1 -*-

# agrega la biblioteca para uso del puerto serie:
import serial
# agrega la biblioteca para uso de funciones
relacionadas con tiempo:
import time

# intenta realizar el bloque de código que le sigue:
try:
    # realiza la conexión con el puerto SERIE
    ser = serial.Serial('/dev/ttyACM0')
    # escribe al puerto serie
    ser.write("led on\n")
    # espera un segundo
    time.sleep(1)
    ser.write("led off\n")
    # cierra la conexión con el puerto serie
    ser.close()
# bloque de código a ejecutar en caso de error
except IOError as Ex:
    print str(Ex)
```

Guarde el archivo con el nombre **serial_write.py** en la carpeta personal del usuario.

2. Analice el código y describa el funcionamiento del programa.
3. Para comprobar el funcionamiento del código: Utilice la terminal de comandos del sistema que abrió anteriormente. Teclee: **python serial_write.py**. Observe y explique el comportamiento del LED conectado a la tarjeta Arduino.

Lectura Serial.

1. Cree un nuevo archivo y en él escriba las siguientes líneas de código.

```
# -*- coding: latin-1 -*-

import serial
try:
    ser = serial.Serial('/dev/ttyACM0')
    # ciclo que ejecuta el código de forma indefinida
    while True:
        # lee desde el puerto serie hasta encontrar un
salto de linea
        # y lo almacena en cmd
        cmd = ser.readline()
        print str(cmd)
```

```
except IOError as e:  
    print str(e)
```

Guarde el archivo con el nombre ***serial_read.py*** en la carpeta personal del usuario.

2. Analice el código y describa el funcionamiento del programa.

Desarrollo de software:

1. Escriba un programa que lea desde el teclado un comando y lo envíe a la tarjeta Arduino a través del puerto serie. Este programa debe ser capaz de encender y apagar el LED de la tarjeta Arduino.
2. Agregue al programa la capacidad de recibir y reconocer la respuesta de la tarjeta Arduino al comando ***magnet***. El programa deberá de imprimir esta respuesta en pantalla.

Práctica 3. Interfaces Web

Objetivo:

- El alumno utilizará herramientas web para implementar una interfaz de usuario básica, utilizando CGI's, que le permita controlar sus actuadores y monitorear el estado de sus sensores.

Introducción

CGI: Interfaz de entrada común (en inglés Common Gateway Interface, abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web.

Desarrollo

Respuestas de CGI's

1. Utilizando el editor de texto **gedit**, abra el archivo ubicado en `/var/www/cgi-bin/sensor.cgi`. Analice el código y describa su funcionamiento.
2. Abra el navegador de internet e ingrese la siguiente dirección <http://localhost/cgi-bin/sensor.cgi>. Posteriormente añada una sentencia de impresión (print) al final del archivo `sensor.cgi` con una cadena de texto cualquiera y salve el documento. Recargue la página en el navegador y compruebe los cambios.
3. Añada las líneas de código necesarias para la conexión con la tarjeta Arduino vía puerto serie; solicite el valor del sensor magnético y reemplace la cadena de texto del punto anterior por dicho valor. Guarde los cambios y recargue la página para comprobar los resultados.

Peticiones a CGI's

1. Utilizando el editor de texto **gedit**, abra el archivo ubicado en `/var/www/cgi-bin/actuator.cgi`. Analice el código y describa su funcionamiento.
2. Agregue las siguientes líneas de código al final del archivo `actuator.cgi`:

```
# almacena todos los datos provenientes del QueryString:  
  
form = cgi.FieldStorage()  
  
# almacena el valor de la variable cmd (proveniente del QueryString)  
  
webCmd = str(form["cmd"].value)  
  
# imprime el valor de webCmd, reflejándose esto en el navegador web  
  
print webCmd
```

3. En el navegador web ingrese a la siguiente dirección http://localhost/cgi-bin/actuador.cgi?cmd=Comando_Arduino y verifique los resultados obtenidos de acuerdo al código anterior.
4. Añada las líneas de código necesarias para la conexión con la tarjeta Arduino vía puerto serie y el envío de la cadena almacenada en la variable *webCmd*. Guarde los cambios.
5. En el navegador web ingrese a la dirección anterior reemplazando "Comando_Arduino" por los comandos de encendido y posteriormente de apagado de LED reconocidos por la tarjeta Arduino.

Paso final:

1. Si realizó todos los pasos anteriores de forma exitosa y respetó todas las convenciones de los comandos; podrá controlar el LED y monitorear el estado del sensor en tiempo real a través de la página web ubicada en la siguiente dirección. <http://localhost/interface.html>.

Nota: De igual forma puede ingresar a las PC's conectadas a la red de la sala sustituyendo "localhost" por la dirección IP. Inclusive desde un móvil, si este se encuentra conectado a la misma red.