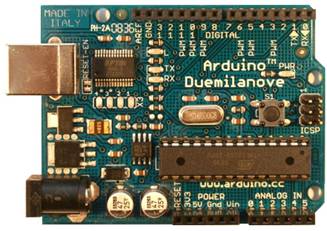
**PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR  
FACULTAD DE INGENIERÍA  
ESCUELA DE SISTEMAS**

**SEMANA 2**

**Nombre:** ERICK PÉREZ  
**Curso:** 6to  
**Tema:** Investigación como programar en Arduino

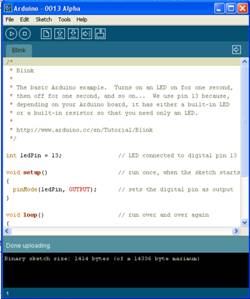
Para programar en arduino es fundamental conocer los componentes físicos que posee dicho dispositivo. Arduino es una placa open hardware por lo que su diseño es de libre distribución y utilización, que incluso podemos construirnos nosotros mismos.



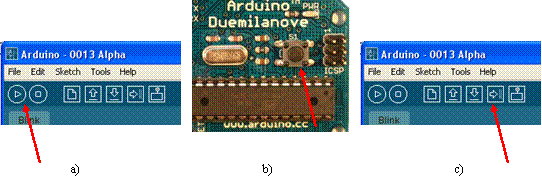
Arduino posee un software para poder transferir el programa directamente a la placa base, el programa se implementará haciendo uso del entorno de programación propio de arduino y se transferirá empleando un cable USB. Si bien en el caso de la placa USB no es preciso utilizar una fuente de alimentación externa, ya que el propio cable USB la proporciona, para la realización de algunos de los experimentos prácticos sí que será necesario disponer de una fuente de alimentación externa ya que la alimentación proporcionada por el USB puede no ser suficiente. El voltaje de la fuente puede estar entre 6 y 25 Voltios.

**Entorno de desarrollo**

Para programar la placa es necesario descargarse de la página web de Arduino el entorno de desarrollo (IDE). Se dispone de versiones para Windows y para MAC, así como las fuentes para compilarlas en LINUX. En la Figura 2 se muestra el aspecto del entorno de programación. En el caso de disponer de una placa USB es necesario instalar los drivers FTDI. Estos drivers vienen incluidos en el paquete de Arduino mencionado anteriormente. Existen en la web versiones para distintos sistemas operativos.



Lo primero que tenemos que hacer para comenzar a trabajar con el entorno de desarrollo de arduino es configurar las comunicaciones entre la placa Arduino y el PC. Para ello deberemos abrir en el menú "Tools" la opción "Serial Port". En esta opción deberemos seleccionar el puerto serie al que está conectada nuestra placa. En Windows, si desconocemos el puerto al que está conectado nuestra placa podemos descubrirlo a través del Administrador de dispositivos (Puertos COM & LPT/ USB Serial Port).



**Estructura básica de un programa**

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

void setup() {

inMode(pin, OUTPUT); // Establece 'pin' como salida

}

void loop() {

digitalWrite(pin, HIGH); // Activa 'pin'

delay(1000); // Pausa un segundo

digitalWrite(pin, LOW); // Desactiva 'pin'

delay(1000);

}

**Funciones**

funció<bookmark>OLE\_LINK1</bookmark>int delayVal() {

int v; // crea una variable temporal 'v'

v = analogRead(pot); // lee el valor del potenciómetro

v /= 4; // convierte los valores 0-1023 a 0-255

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

}

**Variables**

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

int inputVariable = 0;

**Tipos de datos**

Arduino permite manejar los siguientes tipos de datos:

* Byte. Almacena un valor numérico de 8 bits. Tienen un rango de 0-255.
* Int. Almacena un valor entero de 16 bits con un rango de 32,767 a -32,768.
* Long. Valor entero almacenado en 32 bits con un rango de 2,147,483,647 a -2,147,483,648.
* Float. Tipo coma flotante almacenado en 32 bits con un rango de 3.4028235E+38 a -3.4028235E+38.
* Arrays Se trata de una colección de valores que pueden ser accedidos con un número de índice (el primer valor del índice es 0). Ejemplos de utilización:
  + Definición y asignación. int myArray[] = {value0, value1, value2...}
  + Definición. int myArray[5]; // declara un array de 6 enteros
  + Asignación del cuarto componente. myArray[3] = 10;
  + Recuperar el cuarto componente y asignarlo a x. x = myArray[3];

# Bibliografía

Built With Wimba Create. (s.f.). *Manual de programación de Arduino*. Obtenido de http://dfists.ua.es/~jpomares/arduino/page\_01.htm