Arduino – Primera Parte

Introducción

En este primer post mostraremos que es Arduino, como funciona y como cargar el programa Blink. Esto nos permitirá ver el funcionamiento básico y comprobar que nuestra placa funciona correctamente.

- Arduino 

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

El hardware consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, ATmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque (boot loader) que corre en la placa.

Arduino se puede utilizar para desarrollar objetos interactivos autónomos o puede ser conectado a software del ordenador (por ejemplo: Macromedia Flash, Processing, Max/MSP, Pure Data). Las placas se pueden montar a mano o adquirirse. El entorno de desarrollo integrado libre se puede descargar gratuitamente.

Al ser open-hardware, tanto su diseño como su distribución es

libre. Es decir, puede utilizarse libremente para el desarrollo de cualquier tipo de proyecto sin haber adquirido ninguna licencia.

- Processing

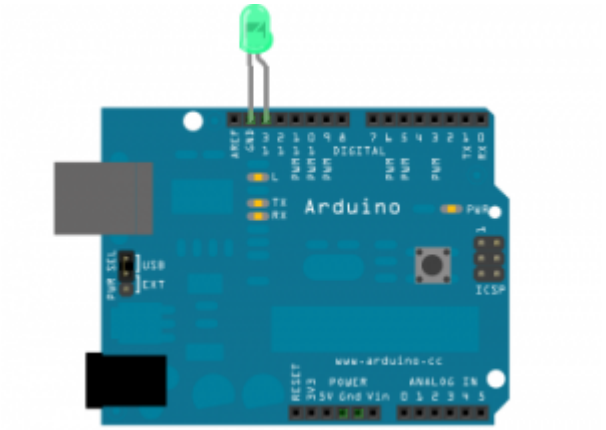


Processing es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en Java, de fácil utilización, y que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital. Fue iniciado por Ben Fry y Casey Reas a partir de reflexiones en el Aesthetics and Computation Group del MIT Media Lab dirigido por John Maeda.
Processing es desarrollado por artistas y diseñadores como una herramienta alternativa al software propietario. Puede ser utilizado tanto para aplicaciones locales así como para aplicaciones web (Applets).
Se distribuye bajo la licencia GNU GPL.

Instalando el software necesario

Deberemos instalar las librerías y el IDE con este simple comando:
sudo apt-get install arduino arduino-core

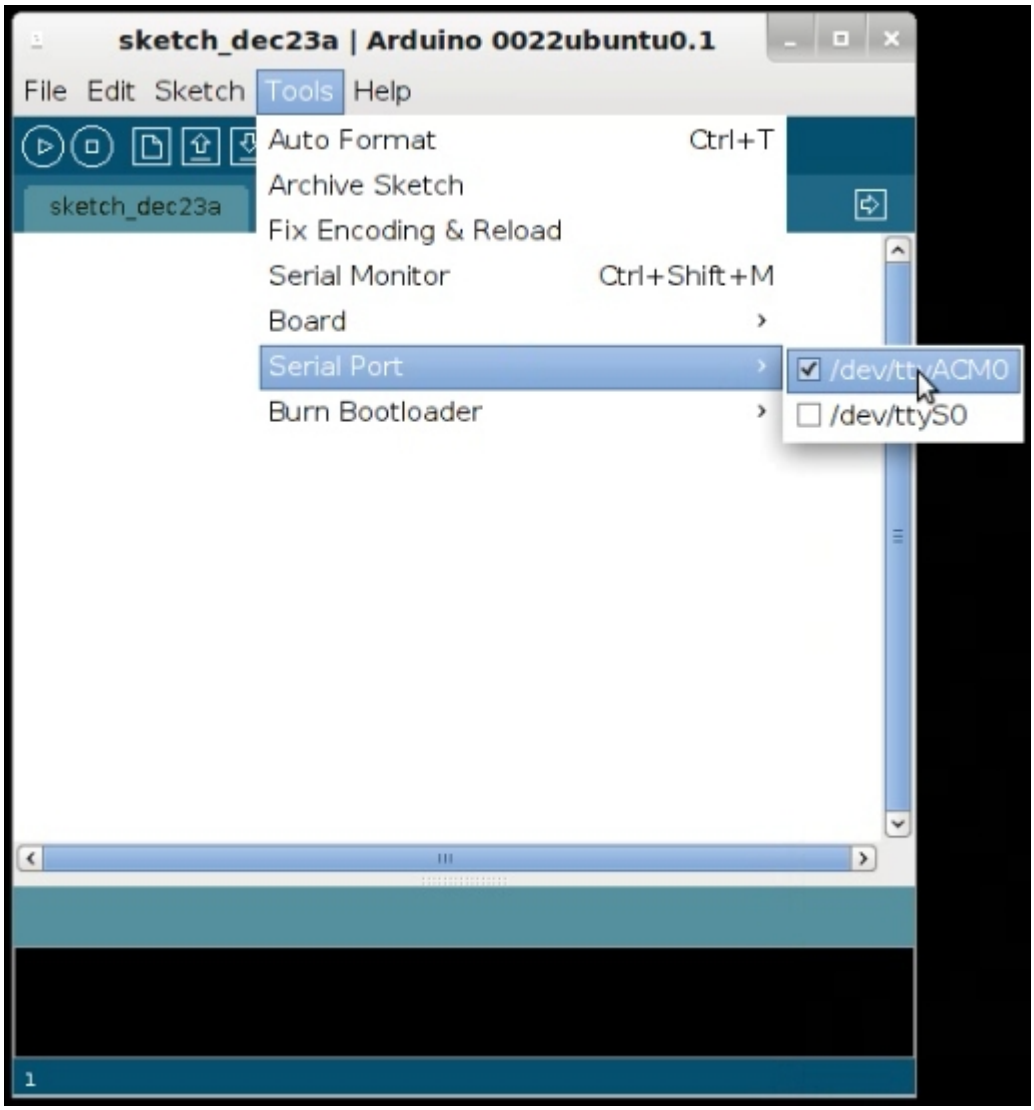
Diseño Arduino



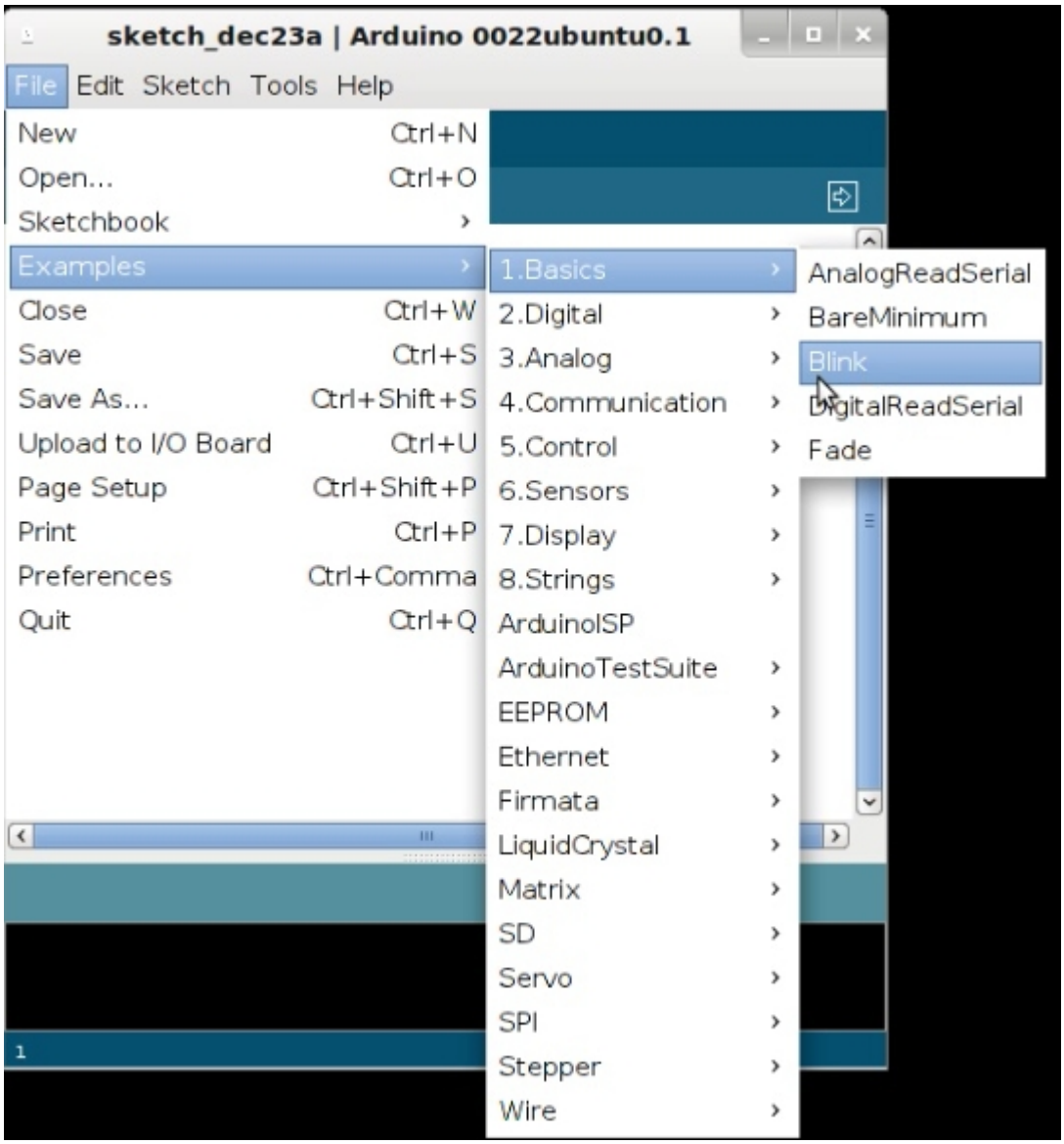
Conectaremos el led teniendo en cuenta que la patilla corta es el negativo y la patilla larga el positivo. Necesitaríamos una resistencia para que no se queme el led, por suerte el pin 13 de la placa Arduino incluye una. Así pues, conectaremos la patilla larga(positivo) al conector 13 y la corta(negativo) a la entrada de corriente GND, como se puede comprobar los dos conectores y la resistencia están puestos estrategicamente.

Cargando el programa Blink

Este programa es el más básico, simplemente produce un parpadeo del led. Conectamos el Arduino por USB. Deberemos comprobar que hemos seleccionado correctamente el puerto USB, en mi caso es el /dev/ttyACM0 tal y como se ve en la imagen.



Cargaremos el programa Blink.



Haremos unas pequeñas modificaciones para poder obtener feedback de lo que esta ocurriendo. Processing nos dirá que el código es read-only, simplemente guardamos el código con un nuevo nombre y listo.

```
/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.
```

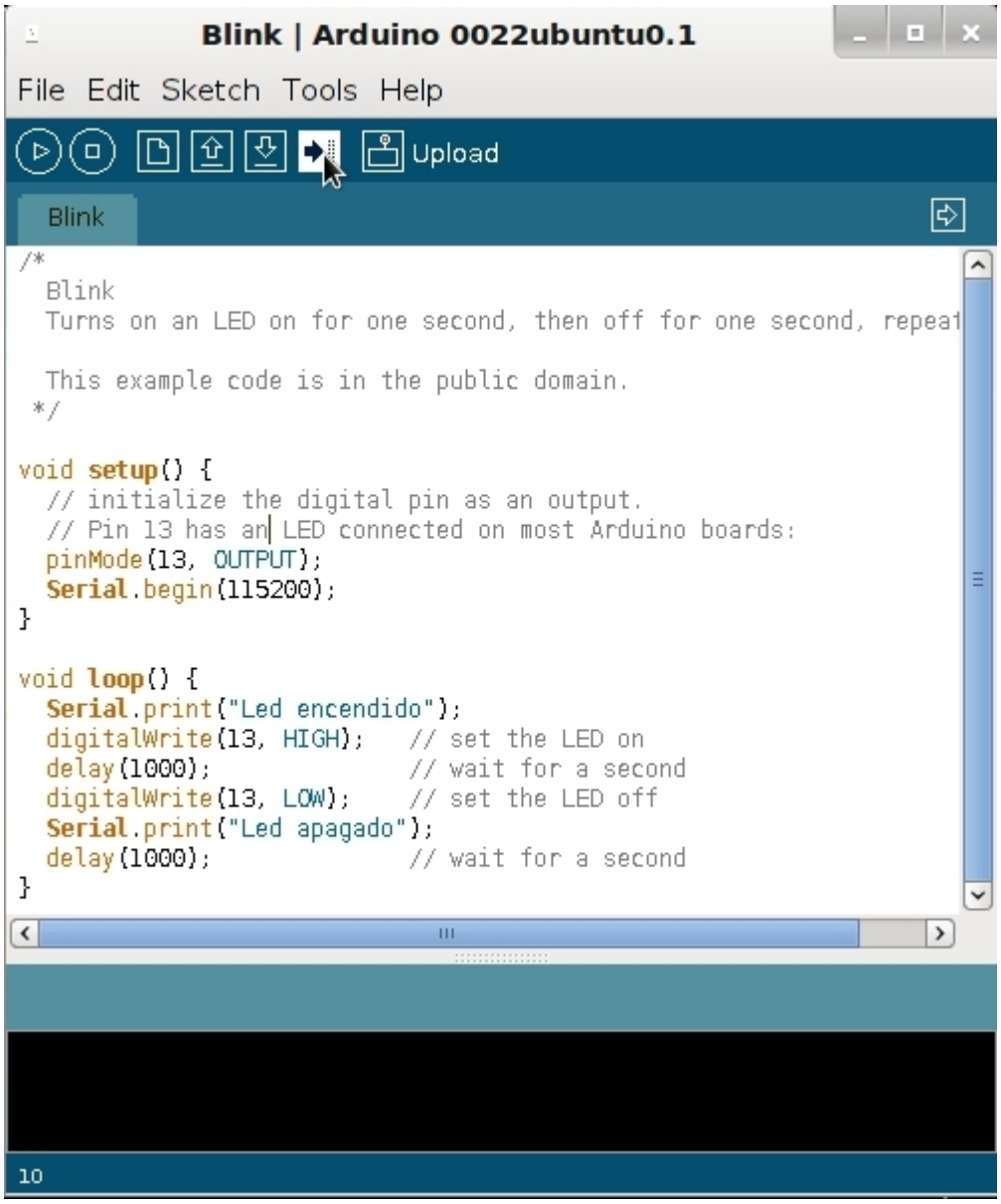
```
This example code is in the public domain.
*/

void setup() {
// initialize the digital pin as an output.
// Pin 13 has an LED connected on most Arduino boards:
pinMode(13, OUTPUT);
Serial.begin(115200); //Establecemos la velocidad de envio de
datos
}

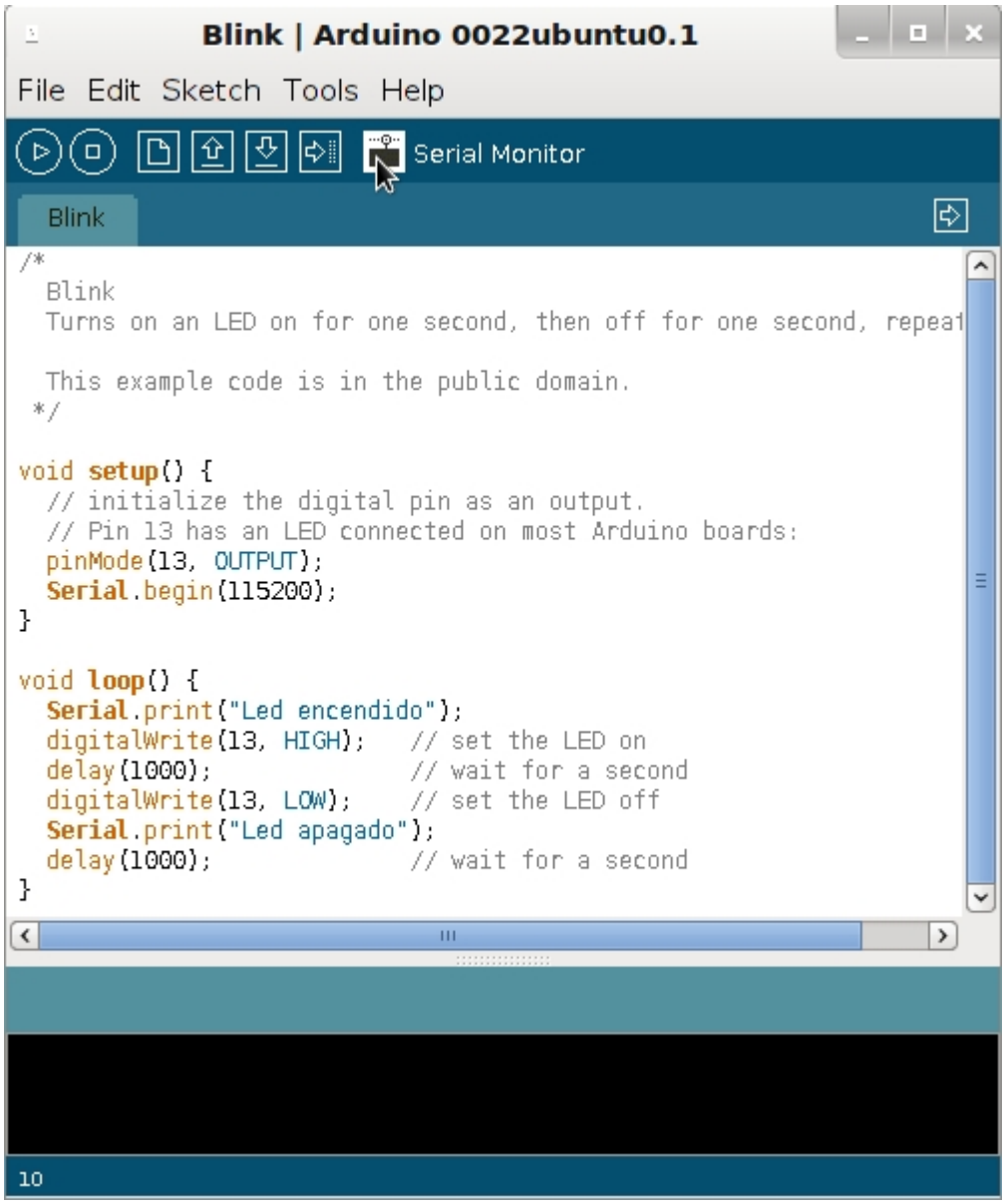
void loop() {
Serial.print("Led encendido");
digitalWrite(13, HIGH); // set the LED on
delay(1000); // wait for a second
Serial.print("Led apagado");
digitalWrite(13, LOW); // set the LED off
delay(1000); // wait for a second
}
```

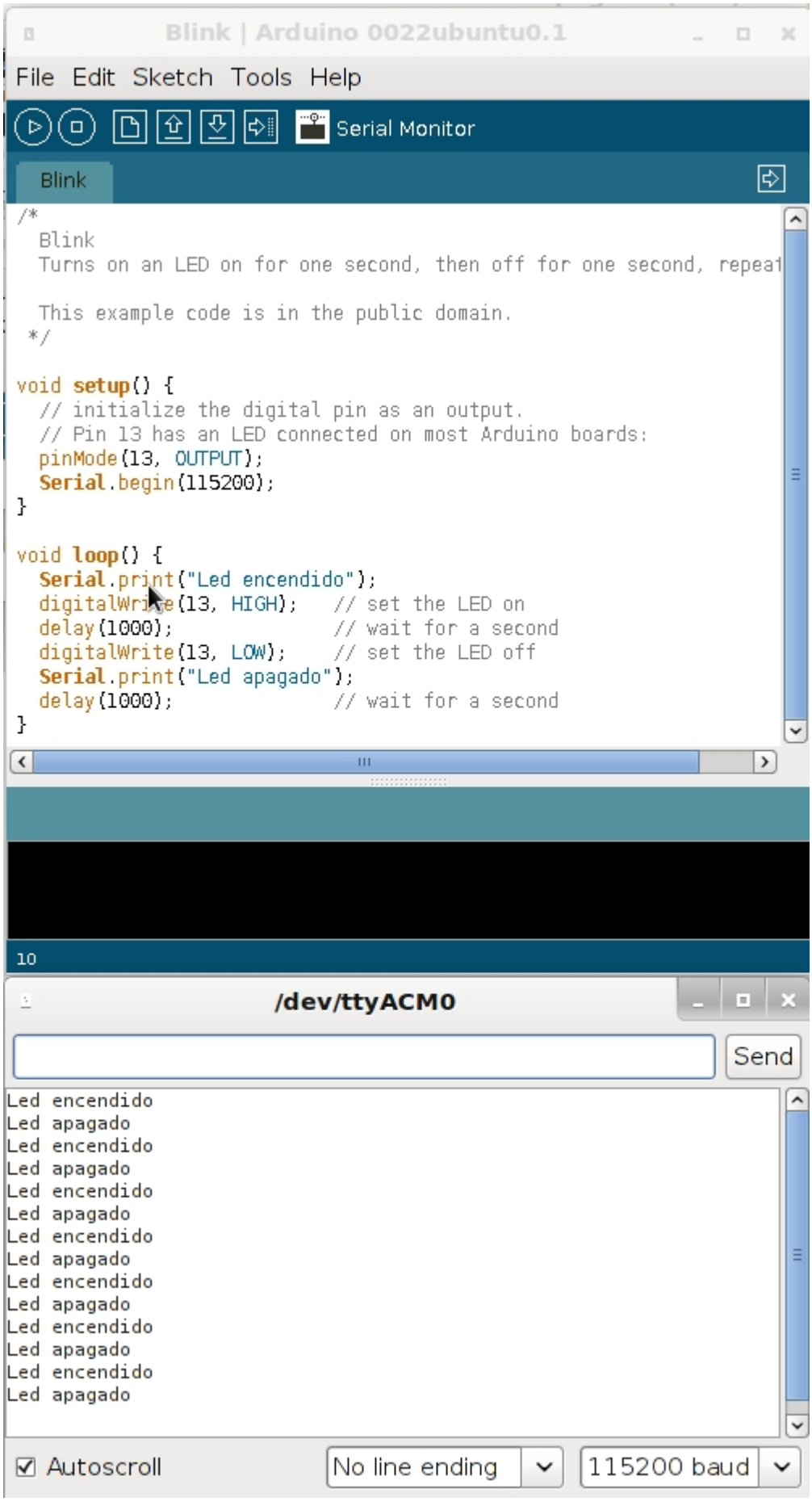
Subir el programa a Arduino y arrancar el Serial Monitor

Subiremos el programa tal y como se ve en la imagen.



Una vez cargado el programa arrancamos el Serial Monitor y podremos ver el feedback que recibimos del Arduino al mismo tiempo que nuestro led empieza a parpadear. Debemos tener en cuenta que la velocidad que hemos indicado en el codigo Serial.begin(115200) sea la misma con la que escucha el Serial monitor(Desplegables inferior derecho).


Blink | Arduino 0022ubuntu0.1
File Edit Sketch Tools Help
Serial Monitor
Blink
/*
Blink
Turns on an LED on for one second, then off for one second, repeat
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);
Serial.begin(115200);
}
void loop() {
Serial.print("Led encendido");
digitalWrite(13, HIGH); // set the LED on
delay(1000); // wait for a second
digitalWrite(13, LOW); // set the LED off
Serial.print("Led apagado");
delay(1000); // wait for a second
}
10


Blink | Arduino 0022ubuntu0.1
File Edit Sketch Tools Help
Serial Monitor
Blink
/*
Blink
Turns on an LED on for one second, then off for one second, repeat
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);
Serial.begin(115200);
}
void loop() {
Serial.print("Led encendido");
digitalWrite(13, HIGH); // set the LED on
delay(1000); // wait for a second
digitalWrite(13, LOW); // set the LED off
Serial.print("Led apagado");
delay(1000); // wait for a second
}
10
/dev/ttyACM0
Send
Led encendido
Led apagado
Led encendido
Led apagado
Led encendido
Led apagado
Led encendido
Led apagado
Led encendido
Led apagado
Led encendido
Led apagado
Led encendido
Led apagado
Autoscroll
No line ending
115200 baud

Observaciones

Con este primer post podemos tener un primer contacto con este espectacular juguete. Las posibilidades son muchas y se pueden encontrar miles de proyectos en la red. Próximamente, mostraré un proyecto un poco más avanzado son Arduino y más adelante el compañero Scuraki continuará con la tercera parte.

Fuentes

- Arduino Blink
- Wikipedia – Arduino

Ruben