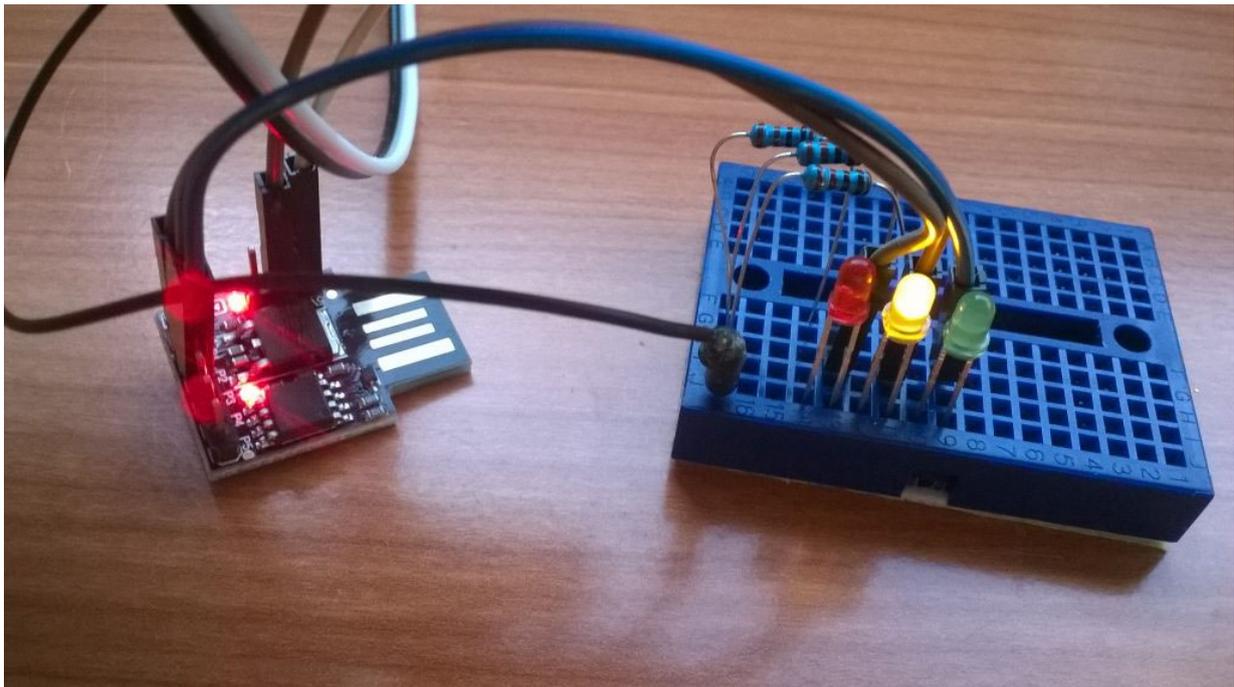
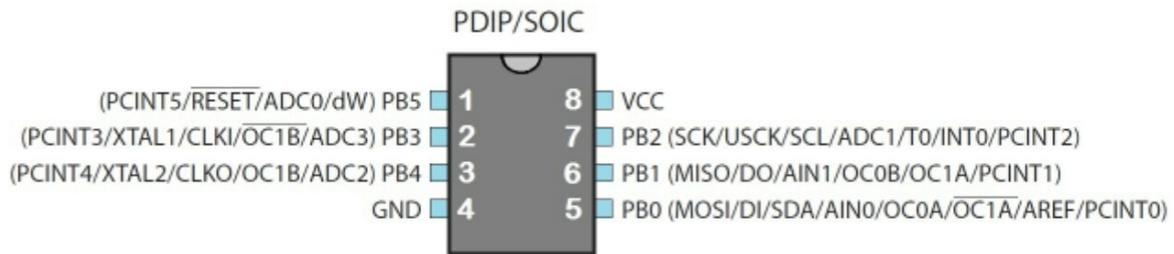


Digispark Attiny USB (un Arduino en miniatura)



El microcontrolador [Attiny85](#) no es muy potente ni demasiado versátil a la vista de los pocos puertos de entrada y salida que tiene, pero es muy pequeño y muy fácil de usar, además de tener un consumo muy bajo y ser muy barato, lo que lo hace perfecto para pequeños proyectos.

Cuando un compañero de trabajo me habló de él no pude esperar a conseguir uno, ya que me encajaba perfectamente para un par de ideas que tenía en la cabeza. Este pequeño microcontrolador de 8 bits sólo tiene 8 patillas: las dos de alimentación y 6 para entrada/salida. Además, lleva un oscilador interno de hasta 20Mhz (si se alimenta a 5,5V), con lo que no es necesario añadir ningún oscilador externo.



Para un primer contacto he usado una pequeña placa de Digispark. Se trata de la [Digispark USB development board](#). Es muy cómoda de usar ya que puede conectarse directamente al puerto USB, tanto para programarla como para alimentarla. También puede alimentarse externamente a través de los tres pines laterales que pueden verse en la fotografía siguiente.

Digispark Attiny USB

Los pines son:

5V – Entrada de alimentación a 5V.

GND – Tierra.

VIN – Entrada de alimentación entre 7V y 12V (la placa dispone de un regulador tipo [7805](#)).

Anuncio publicitario



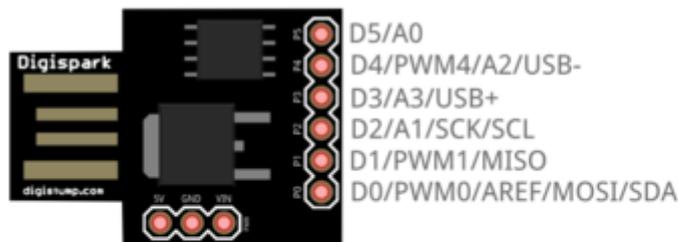
Build Online Store

1. Choose Hosting Provider 2. Install WordPress 3. Select Theme 4. Install eCommerce Plugin...

[➤](#)

[Ajustes de privacidad](#)

Internamente dispone de 8Ks de memoria (aunque el bootloader necesita 2 de esos Kilobytes), y de los seis pines de entrada/salida, además de como E/S digitales, tres pueden ser usados como PWM y 4 como ADC.



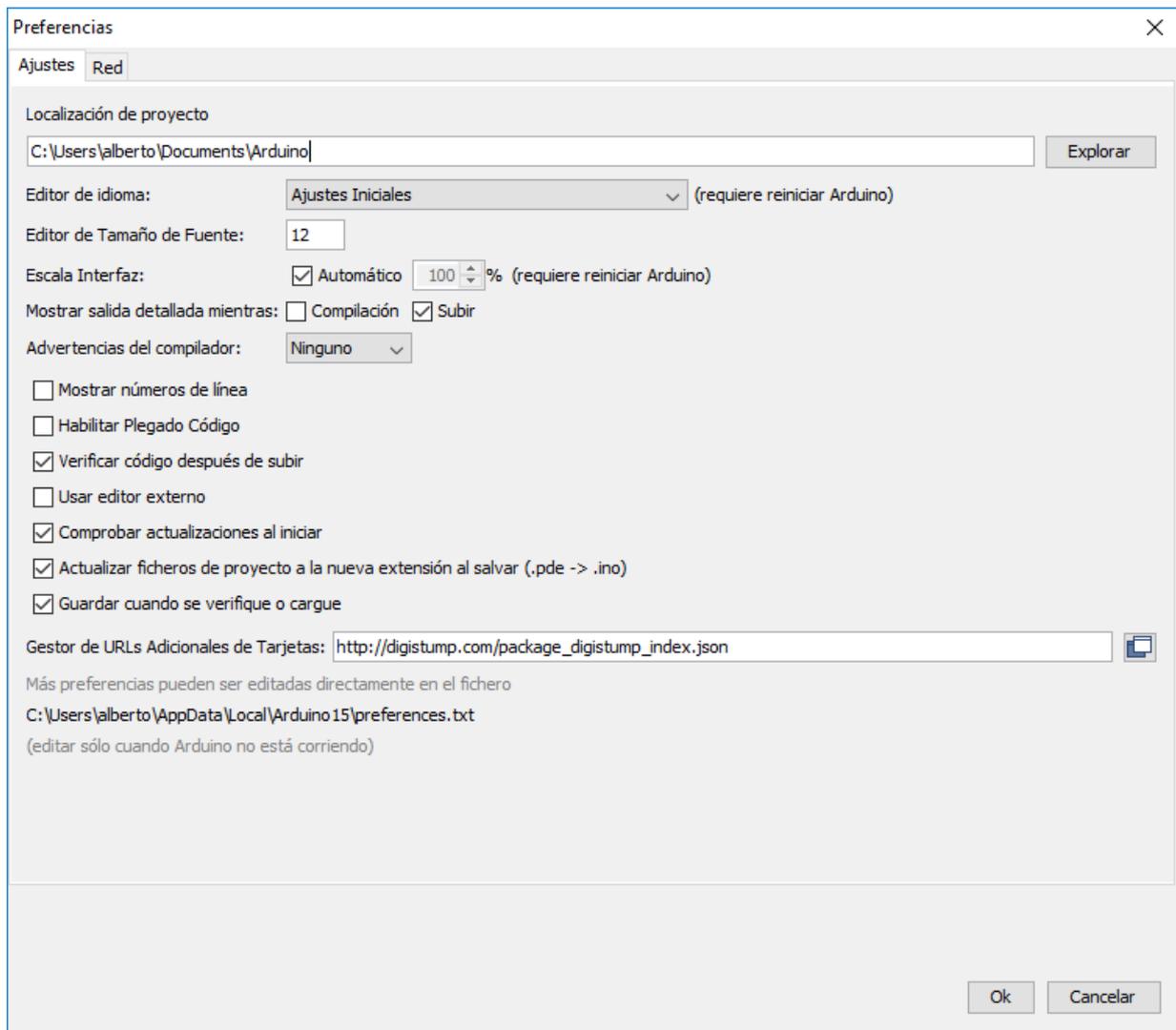
D: Digital Read/Write
A: Analog Read (ADC)
PWM: Analog Write
MOSI/MISO/SCK: SPI
SDA/SCL: I2C
USB+/-: USB Interface
AREF: Analog Reference

fritzing

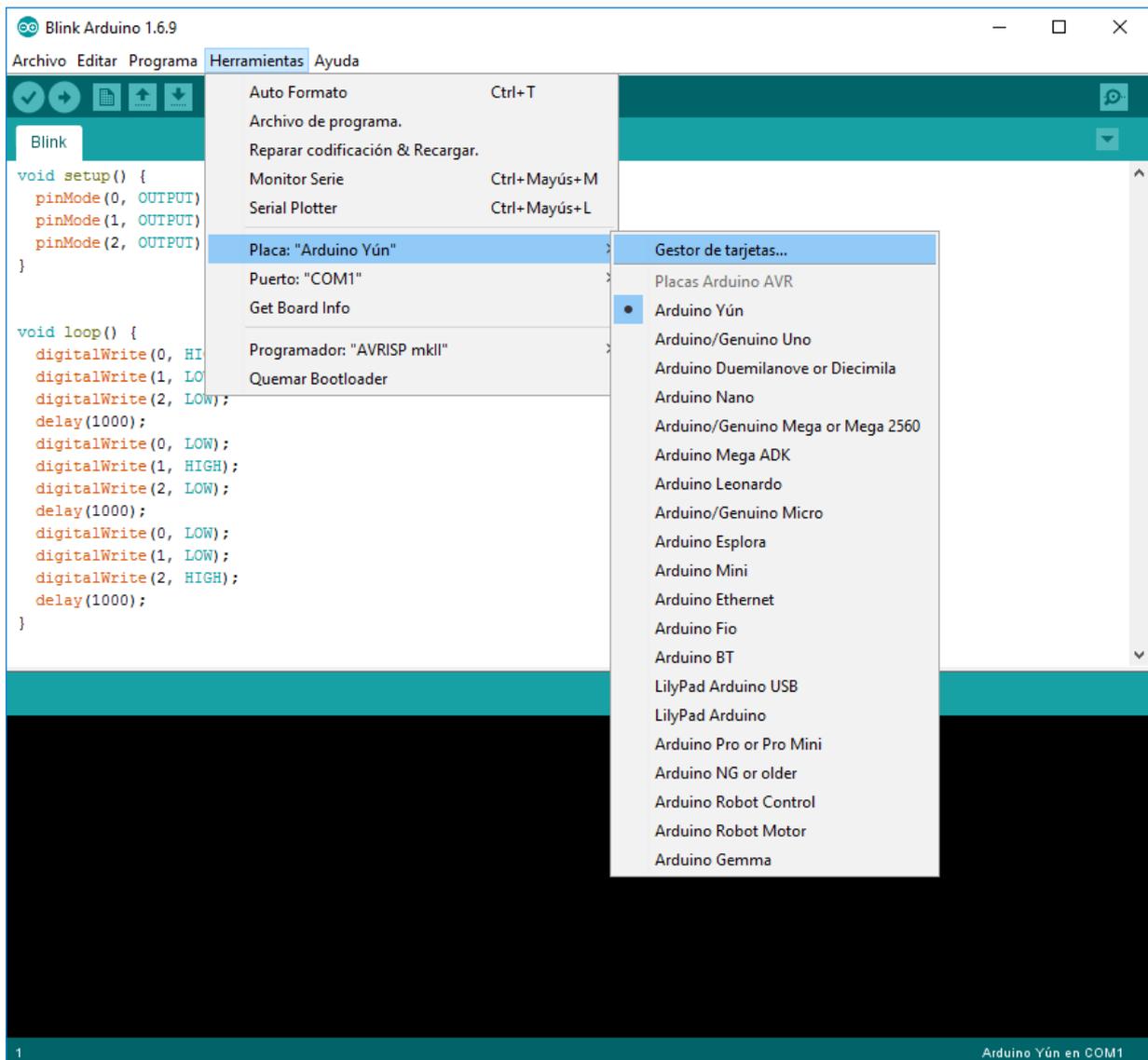
Esta pequeña placa se puede programar desde el entorno de desarrollo de Arduino, por lo que podremos adaptar desarrollos previos que hayamos hecho para otros microcontrolares soportados por esta plataforma. Los pasos para poder usar esta placa en el IDE de Arduino son las siguientes.

Primero hay que descargar e instalar los *drivers* de la placa desde <https://github.com/digistump/DigistumpArduino/releases/download/1.6.7/Digistump.Drivers.zip>.

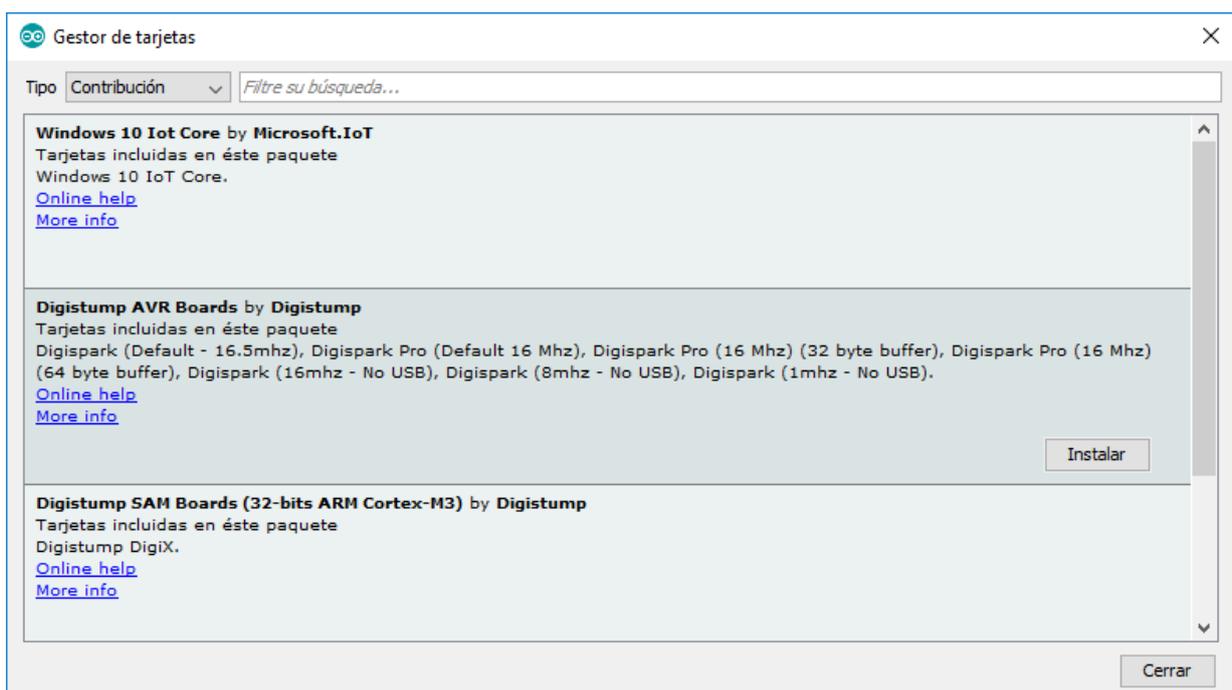
Una vez instalados, vamos a *Archivo > preferencias* y en el campo *Gestor de URLs Adicionales de Tarjetas* ponemos la siguiente URL: http://digistump.com/package_digistump_index.json



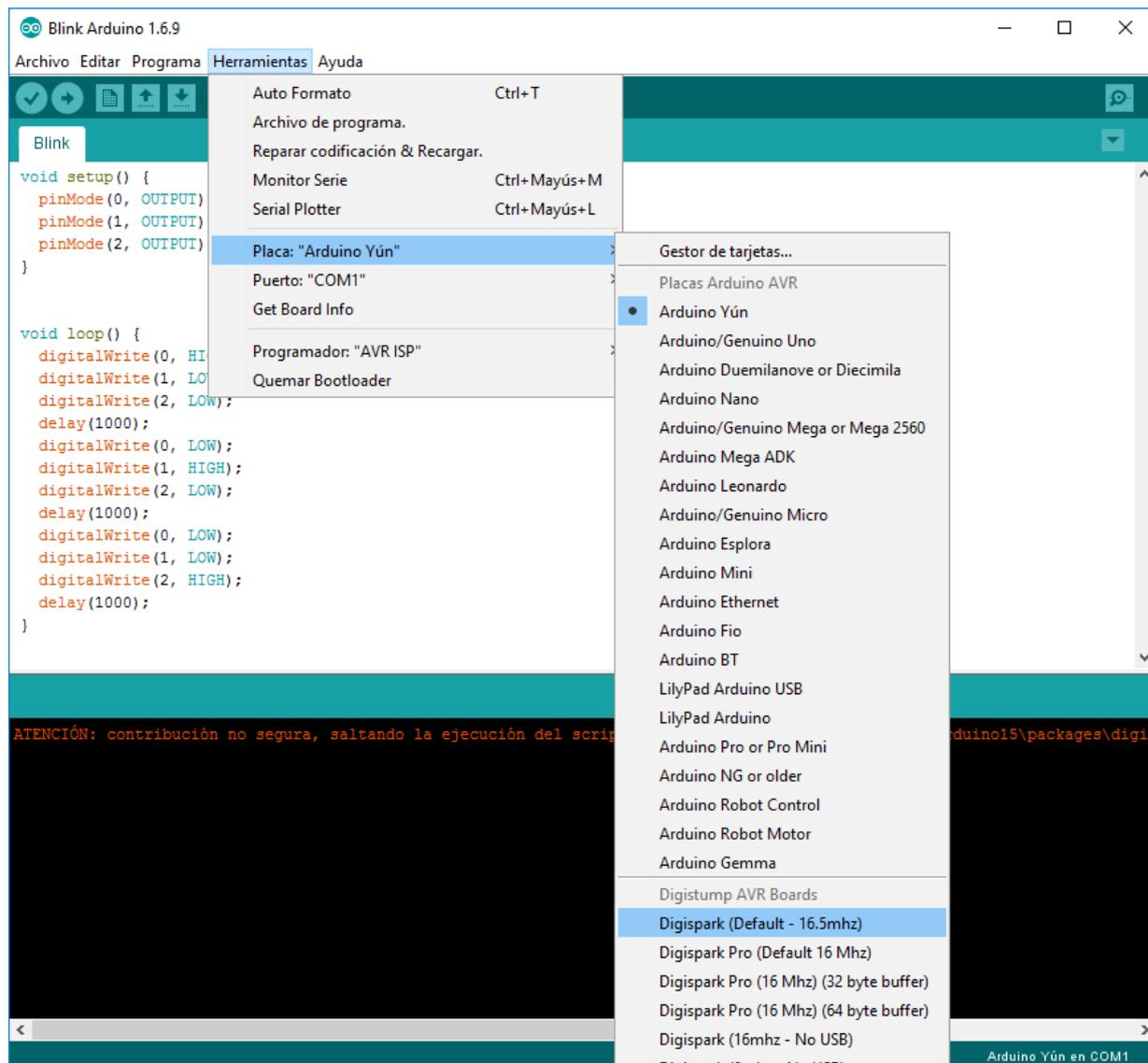
En el menú *Herramientas>Placa* seleccionamos *Gestor de Tarjetas...*



Buscamos la placa *Digistump AVR* y pulsamos el botón *Instalar*.

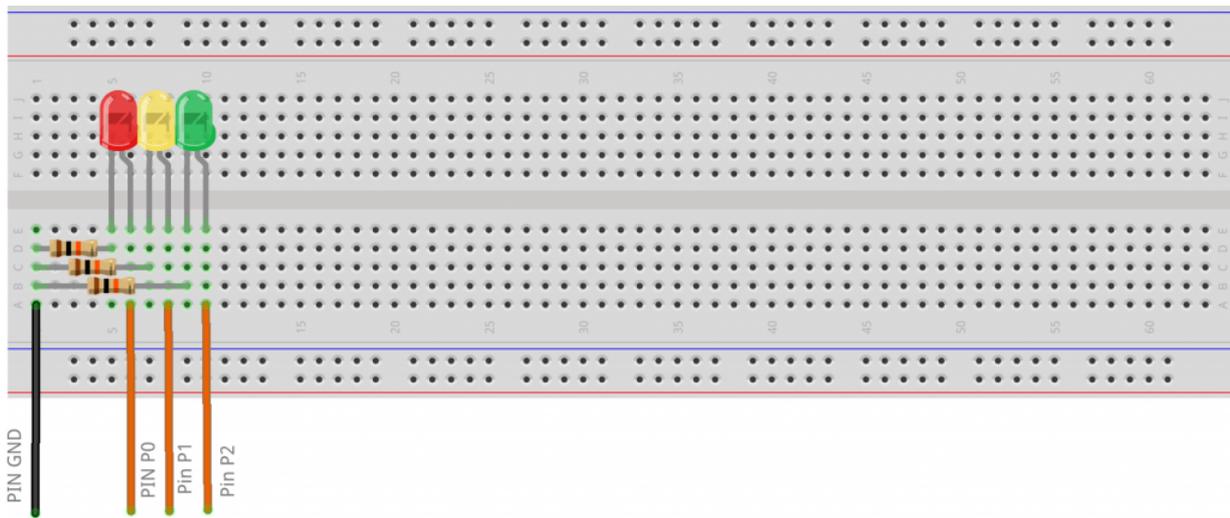


Elegimos la placa *Digispark (Default – 16.5Mhz)* en el menú *Herramientas>Placa*.



Nos aseguramos de que en *Herramientas>Programador* está seleccionado *AVRISP mskII*. En principio, con esto, está todo listo para empezar a trabajar con la placa. Vamos a hacer un pequeño montaje muy sencillo para ponerla a prueba.

Vamos a reutilizar el mismo montaje del [artículo anterior](#), esta es la distribución de los elementos en la protoboard.



fritzing

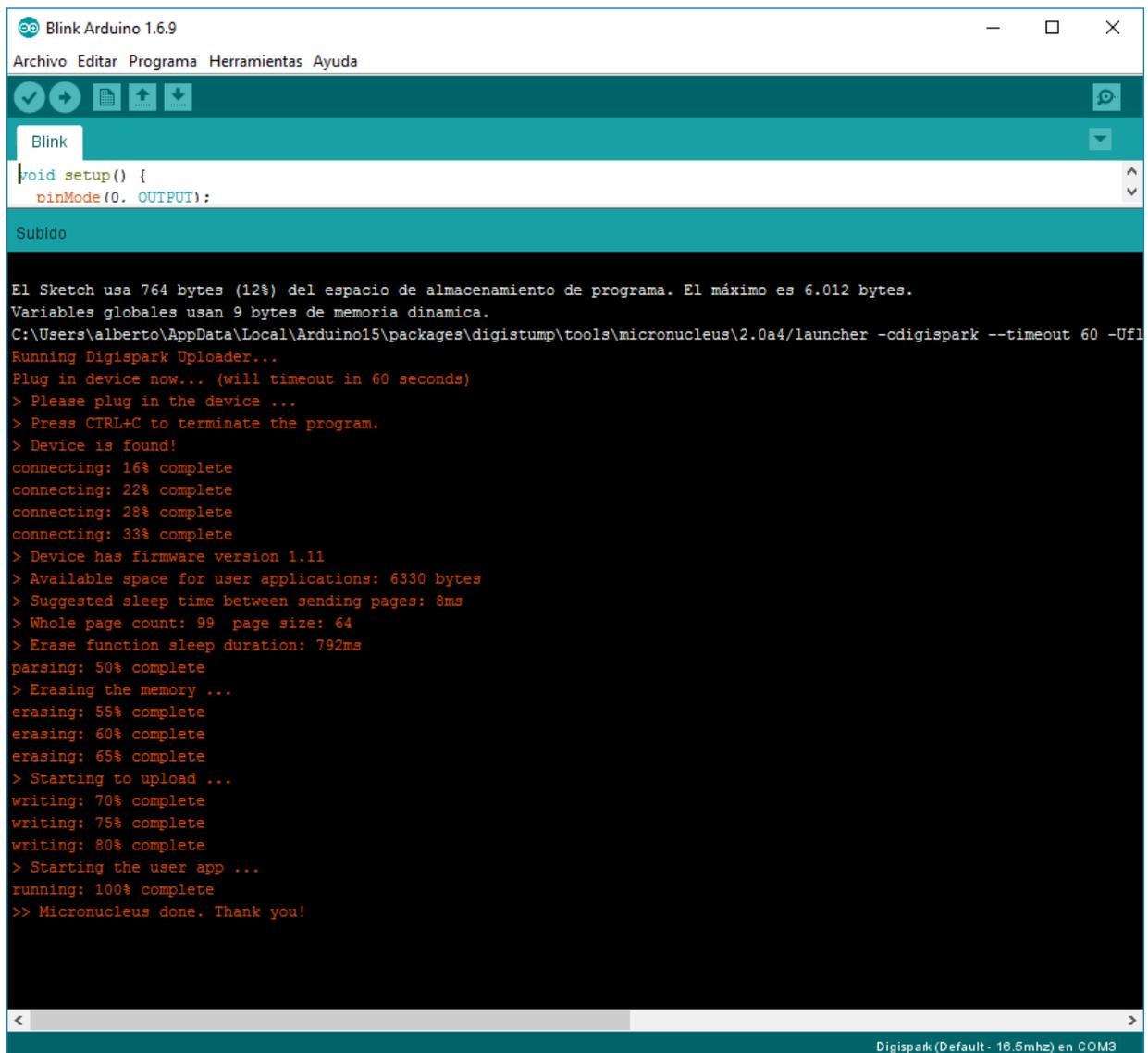
Queremos ir encendiendo alternativamente cada led a intervalos de un segundo, así que conectaremos los leds a los pines 0, 1 y 2 de la placa. El programa que vamos a cargar en el Attiny85 es el siguiente.

```
void setup() {
  pinMode(0, OUTPUT);
  pinMode(1, OUTPUT);
  pinMode(2, OUTPUT);
}

void loop() {
  digitalWrite(0, HIGH);
  digitalWrite(1, LOW);
  digitalWrite(2, LOW);
  delay(1000);
  digitalWrite(0, LOW);
  digitalWrite(1, HIGH);
  digitalWrite(2, LOW);
  delay(1000);
  digitalWrite(0, LOW);
  digitalWrite(1, LOW);
  digitalWrite(2, HIGH);
  delay(1000);
}
```

Antes de subir el programa al micro, lo **desconectamos** del puerto USB y pulsamos la flechita en el IDE para programar la placa. Ahora el programador está a la espera de que conectemos la placa al puerto USB (ahora sí la conectamos). Cuando la detecta, sube el programa y ya podemos volver a

desconectarla.



```
Arduino IDE - Blink Arduino 1.6.9
Archivo Editar Programa Herramientas Ayuda
Blink
void setup() {
  pinMode(0, OUTPUT);
}
Subido
El Sketch usa 764 bytes (12%) del espacio de almacenamiento de programa. El máximo es 6.012 bytes.
Variables globales usan 9 bytes de memoria dinamica.
C:\Users\alberto\AppData\Local\Arduino15\packages\digistump\tools\micronucleus\2.0a4\launcher -cdigispark --timeout 60 -Uf1
Running Digispark Uploader...
Plug in device now... (will timeout in 60 seconds)
> Please plug in the device ...
> Press CTRL+C to terminate the program.
> Device is found!
connecting: 16% complete
connecting: 22% complete
connecting: 28% complete
connecting: 33% complete
> Device has firmware version 1.11
> Available space for user applications: 6330 bytes
> Suggested sleep time between sending pages: 8ms
> Whole page count: 99 page size: 64
> Erase function sleep duration: 792ms
parsing: 50% complete
> Erasing the memory ...
erasing: 55% complete
erasing: 60% complete
erasing: 65% complete
> Starting to upload ...
writing: 70% complete
writing: 75% complete
writing: 80% complete
> Starting the user app ...
running: 100% complete
>> Micronucleus done. Thank you!
Digispark (Default - 16.5mhz) en COM3
```

La placa se puede alimentar conectándola a un puerto USB o externamente. En el siguiente vídeo se ve el programa que acabamos de subir funcionando (en este caso con alimentación externa desde una fuente de alimentación a 5V).