



Arduino Inicial

Ing. Juan C. Abdala

Clase 1



Objetivo de esta clase

- Aprender qué es Arduino
- Aprender a crear un proyectos con Arduino
- Aprender a enviar señales digitales.
- Aprender a usar el puerto serial.



Modalidad

1. Teorico 1 h 30 min
2. Descanso 15 min
3. Practico 1h 15 min

Material: <https://github.com/jcabdala/ArduinolnicialUNC>



Presentacion

¿Quienes somos?

¿Que sabemos? de:

- *Electronica

- *Programacion

¿Que queremos aprender?



Más preguntas

¿Qué crees que es Arduino?



Cronograma

1. Repaso conceptos de electrónica.
2. Introducción a Arduino.
3. Nuestro primer Proyecto.



Formato

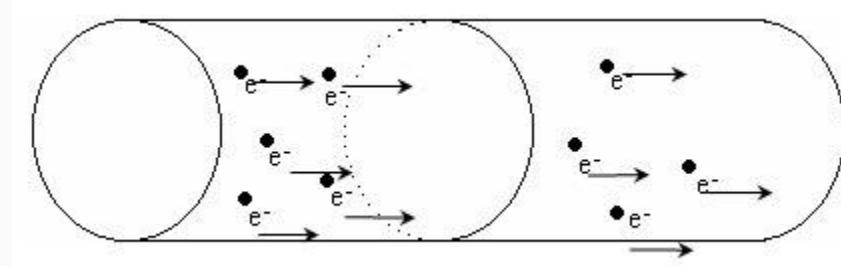
**Decir, hacer y preguntar
(ómo Arduino mismo).**



Intro. Electronica

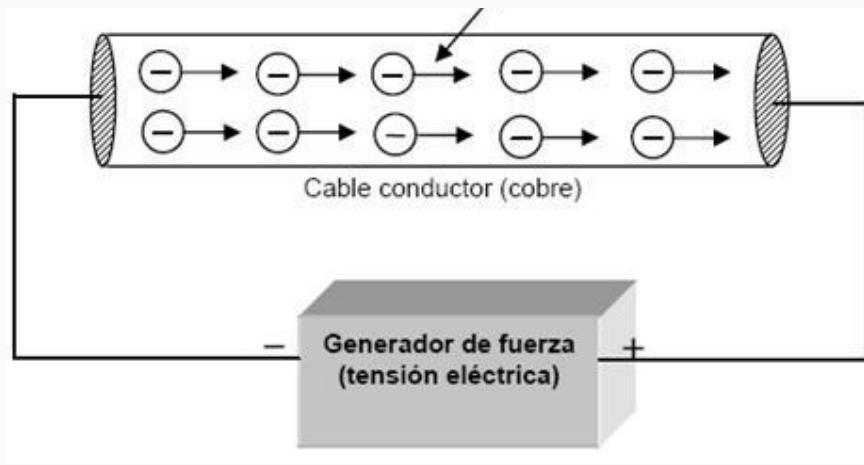
Ley de Ohm

Corriente (I) [Ampere]: flujo de carga eléctrica por unidad de tiempo que recorre un material.



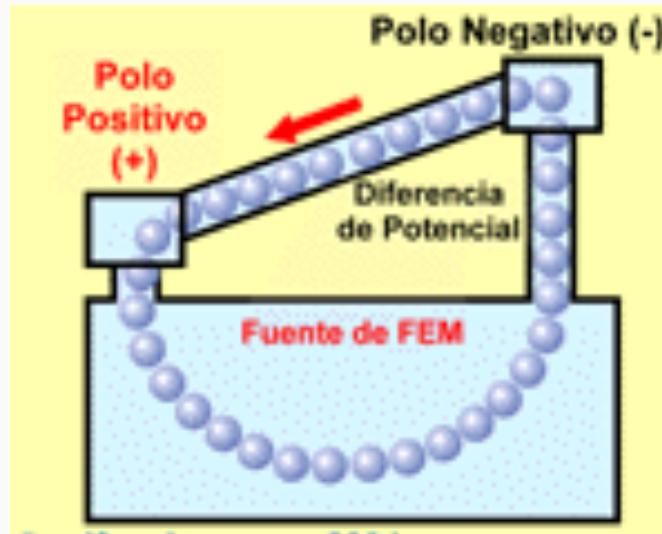
Ley de Ohm

Tensión (V) [volt]: es una magnitud física que cuantifica la diferencia de potencial eléctrico entre dos puntos.



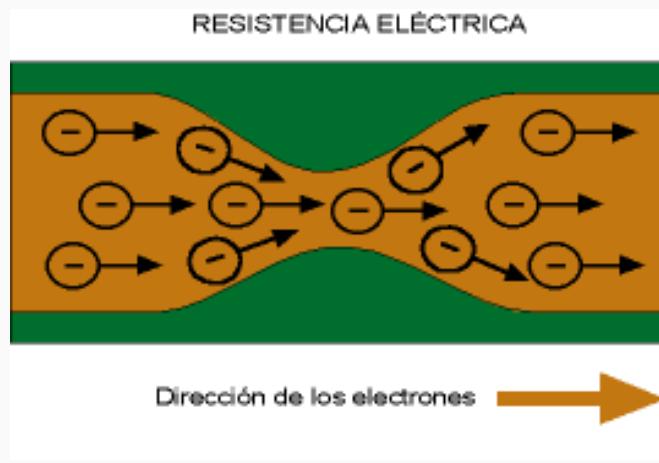
Ley de Ohm

Tensión ¿diferencia de potencial??????



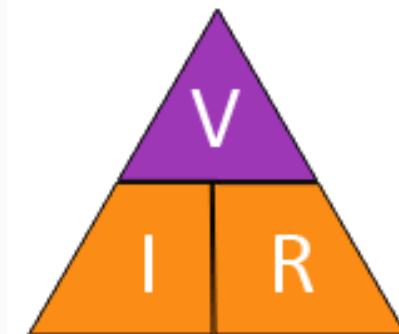
Ley de Ohm

Resistencia(R) [Ω]: Oposición que presenta un conductor al paso de la corriente eléctrica.



Ley de Ohm

“La corriente eléctrica es directamente proporcional al voltaje e inversamente proporcional a la resistencia eléctrica.”

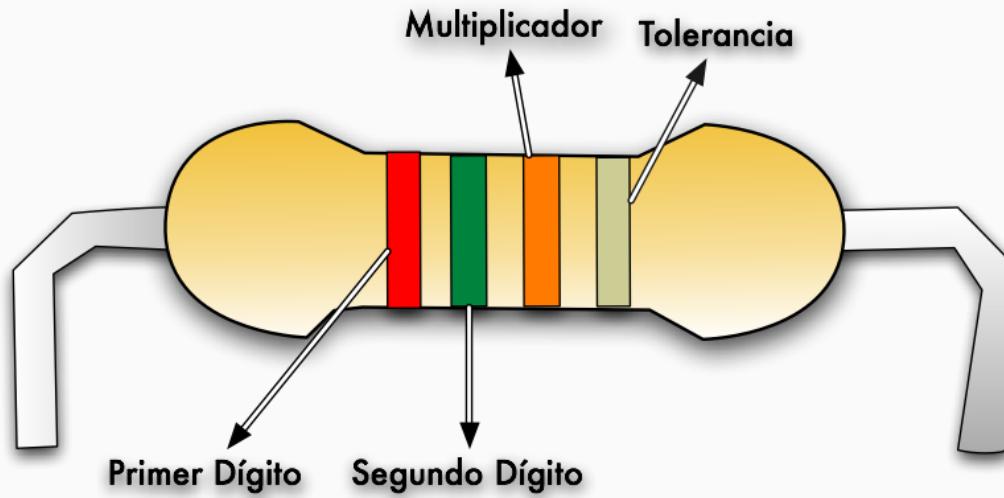


$$V = I * R$$
$$I = V / R$$
$$R = V / I$$



Componentes Básicos.

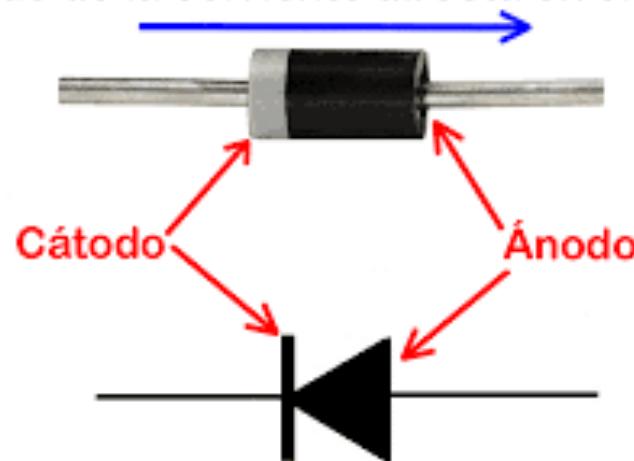
Resistencia.



Componentes Básicos.

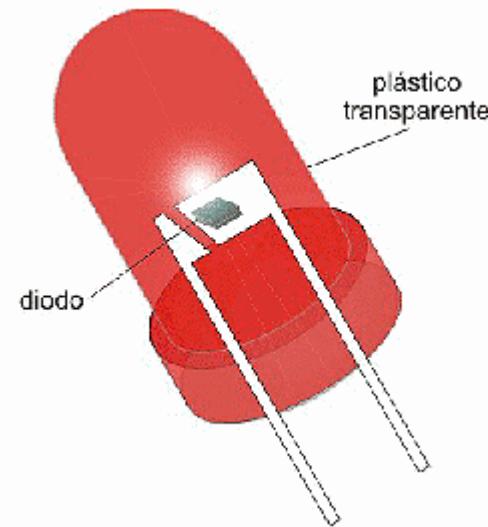
Diodo:

Sentido de la corriente directa en el diodo



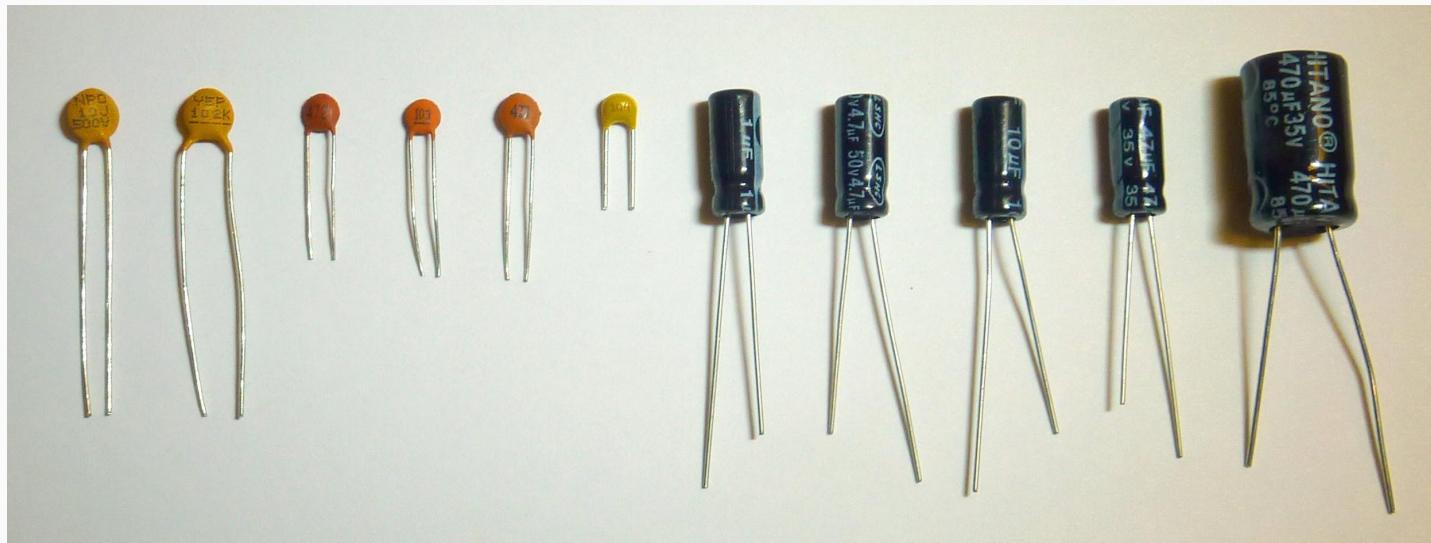
Componentes Básicos.

Diodo LED:



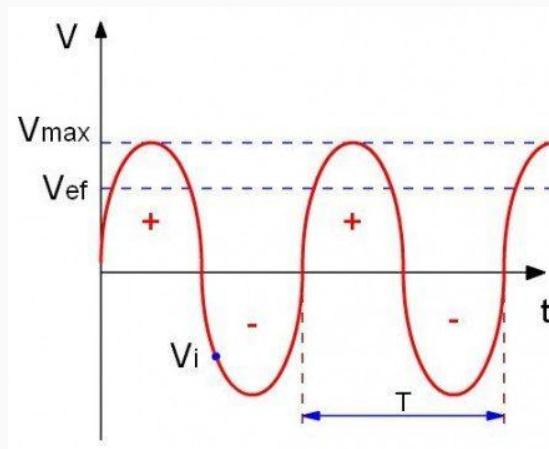
Componentes Básicos.

Capacitor [Faradio]:



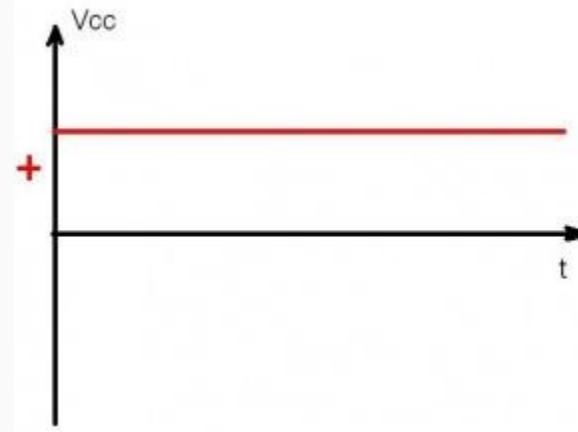
Corriente Alterna

Corriente eléctrica variable en la que las cargas eléctricas cambian el sentido del movimiento de manera periódica (AC).



Corriente Continua

Corriente de intensidad constante en la que el movimiento de las cargas siempre es en el mismo sentido (DC).



Fuentes Rectificadores

Disminuye Tensión, transforma de AC a DC y estabiliza en un valor fijo.



Estabilizadores de tensión

Un regulador de tensión o regulador de voltaje es un dispositivo electrónico diseñado para mantener un nivel de tensión constante.



Multimetro o Tester



Multimetro o Tester

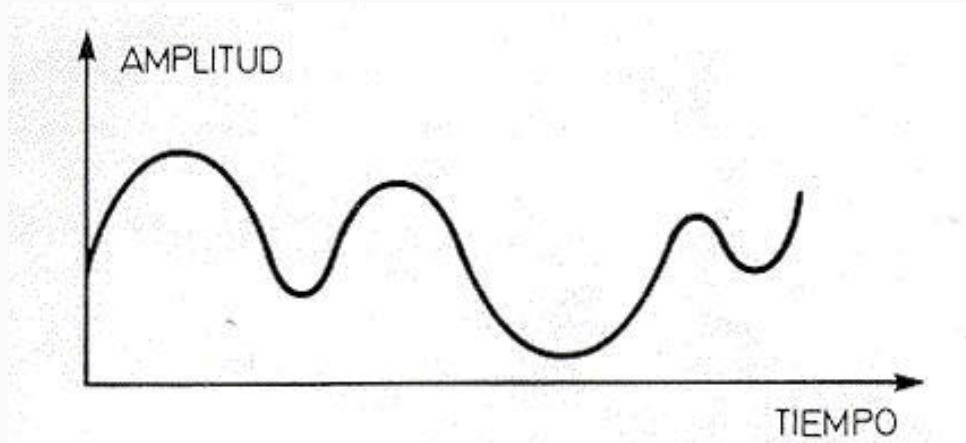
Mide:

- Tension continua. !!
- Tension alterna.
- Resistencia. !!
- Corriente. !!
- Capacidad.
- Continuidad. !!



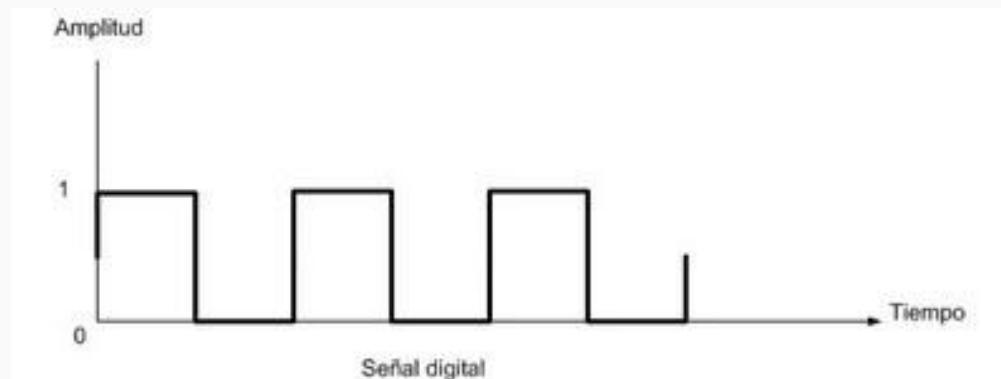
Señal Analógica

Una señal analógica es continua, y puede tomar infinitos valores de Tensión (DC).



Señal Digital

Usan la lógica de dos estados representados por dos niveles de **Tensión (DC)** eléctrica, uno alto, H y otro bajo, L (de High y Low).



Electronica Digital.



Números Binarios

Sistema de numeración en el que los números se representan utilizando solamente las cifras cero y uno (0 y 1).

10101010
[8bits]



Números Binarios

Método de cálculo: consiste en valuar las posiciones en potencias sucesivas de 2 de modo que su suma resulte ser el número decimal a convertir.



Números Binarios

la séptima posición en el número

posición cero en el número

sistema de numeración en base 2

$$11011010 = 1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$$

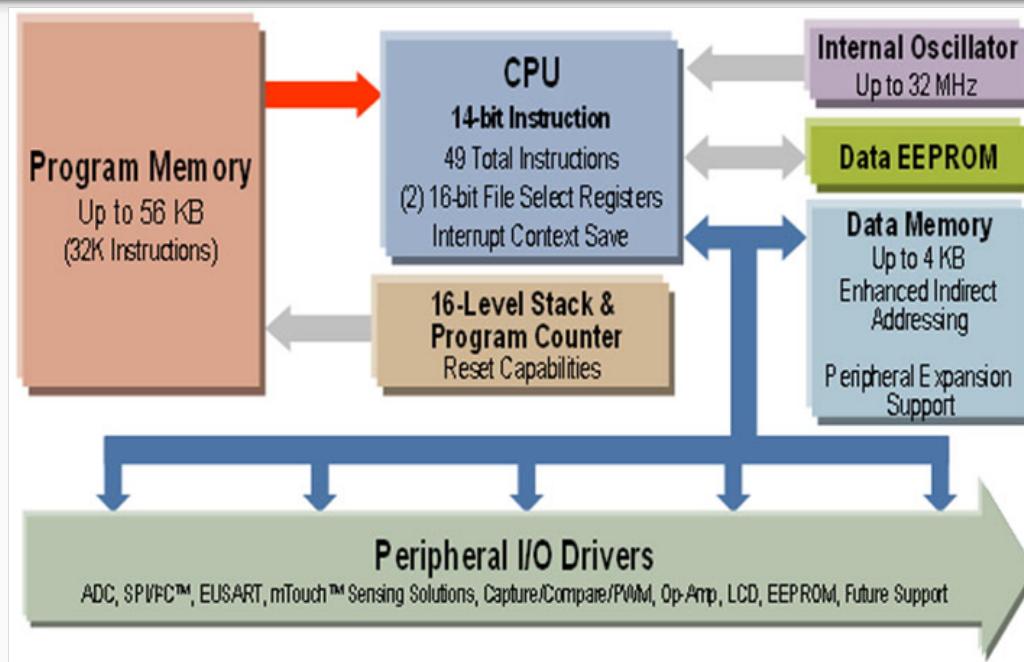
$$11011010 = 128 + 64 + 0 + 16 + 8 + 0 + 2 + 0 = 218$$

Número 218 en el sistema binario

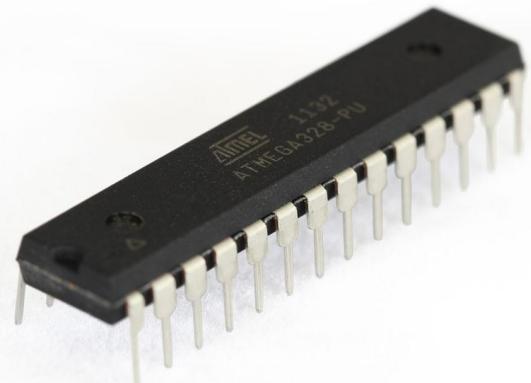
El mismo número en el sistema decimal



Microcontrolador



Microcontrolador



¿Como se usa?

- Se debe construir o comprar una placa de desarrollo.
- Se debe estudiar su manual (datasheet) para saber como configurarlo.
- Se debe configurar a nivel código según su uso.
- Con cada cambio de código se debe grabar.



¿Como se usa?

- Se debe instalar y usar el entorno de desarrollo del fabricante.
- Se pueden usar el IDE del fabricante, la mayoría de las empresas solo soportan windows.
- LA EMPRESA NO PROPORCIONA LIBRERÍAS PARA COMPONENTES DEL MICRO.



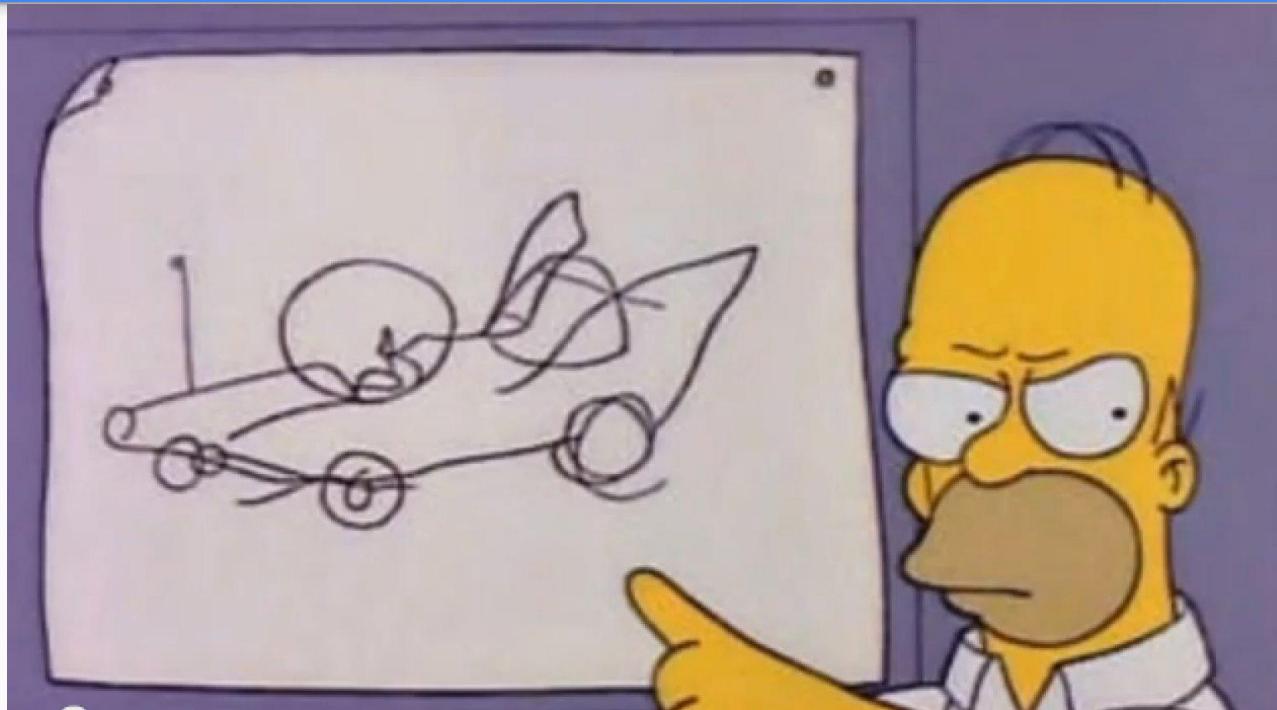
¿Qué es un sistema embebido?

- Diseñados para cubrir necesidades específicas.
- La mayoría de los componentes se encuentran incluidos en la placa base.
- Ejemplos:
Taxímetros, control de acceso, cajeros, fotocopiadora...

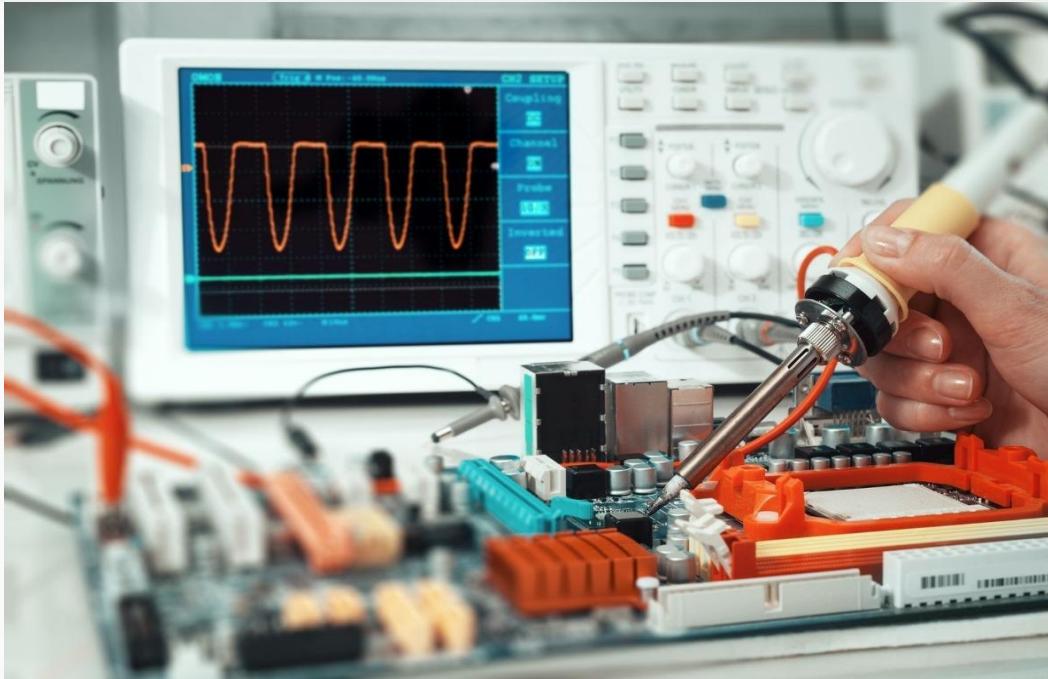


¿Por qué existe Arduino?

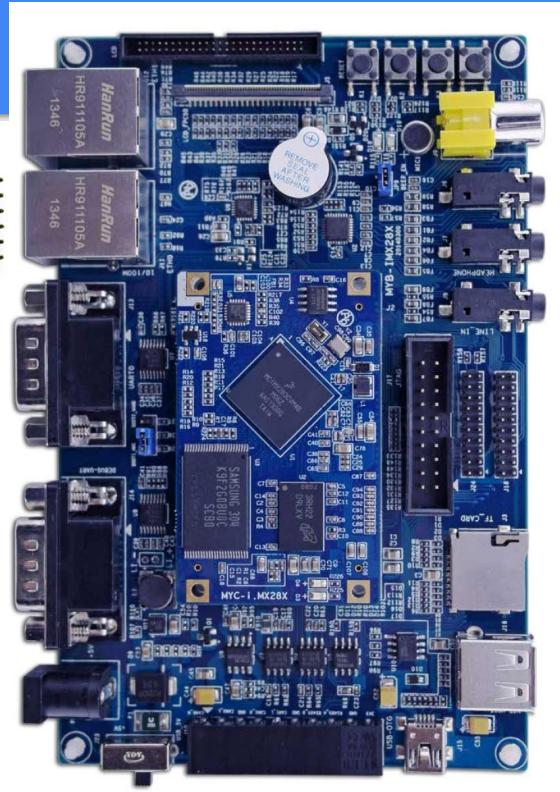
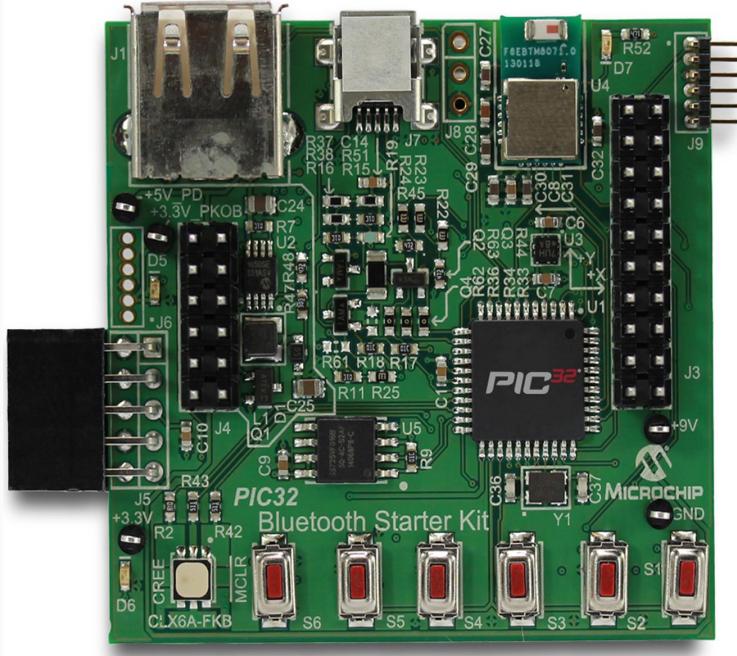
El proceso de desarrollo de hardware



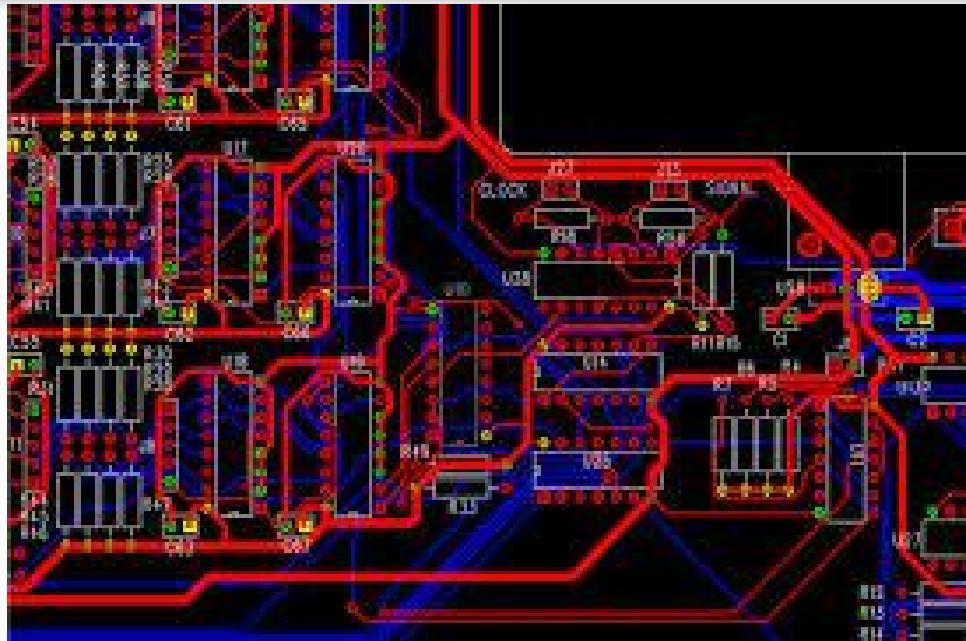
Personal Super capacitado



\$\$\$\$\$\$\$\$\$



Diseño de placas



Software

*Directo con ASM, C o C++

*Embebemos un sistema, Linux o Free BSD y programamos un driver o funcionalidad.



Mucho Tiempo == \$\$\$



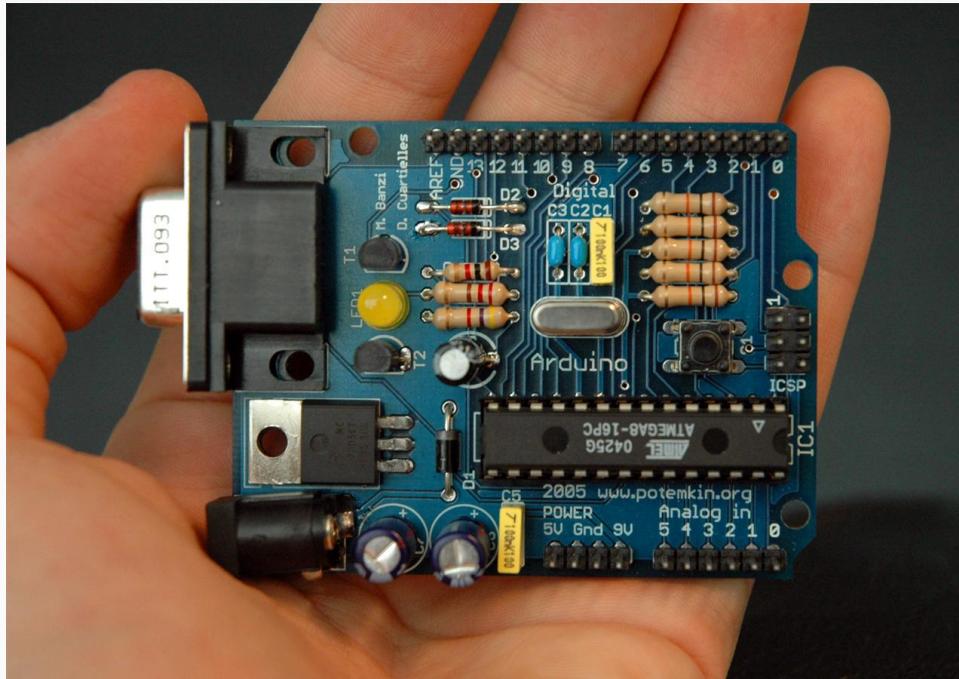
© www.123rf.com



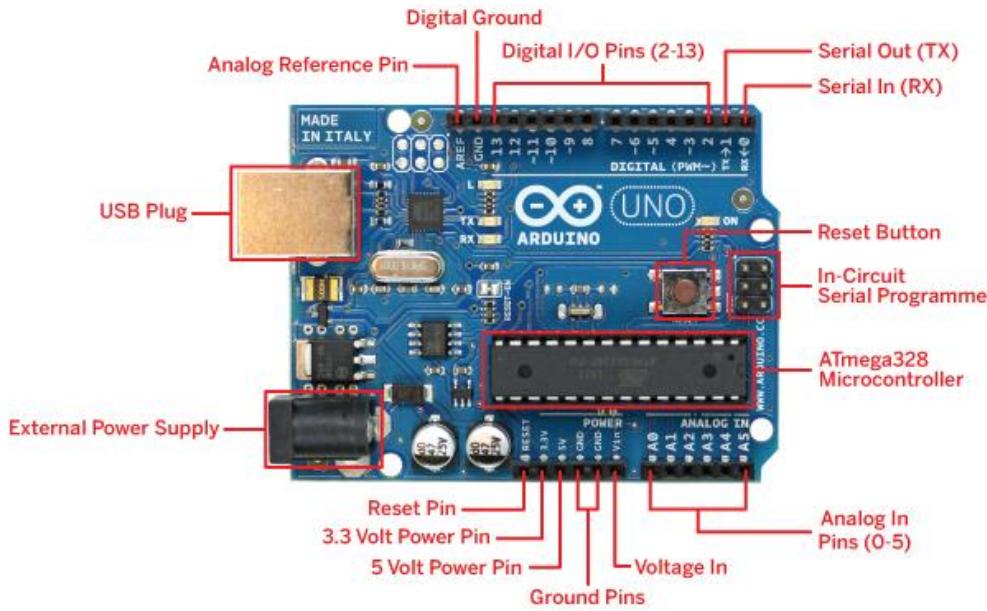
Creadores de Arduino



Arduino - Bootloader



Arduino



Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

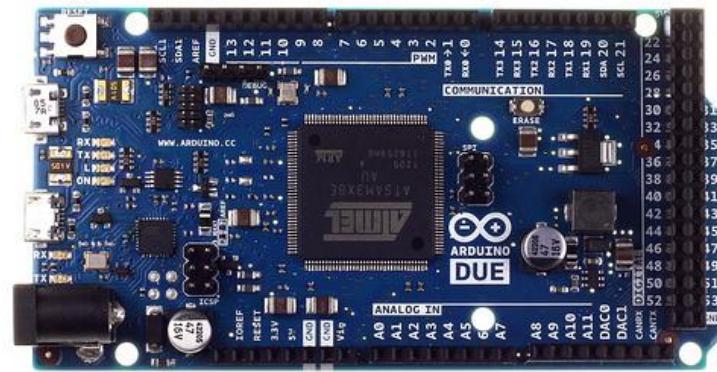
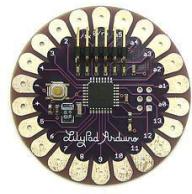


Ecosistema Arduino

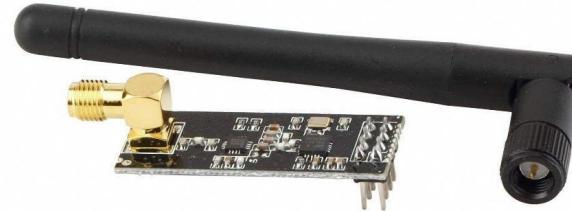
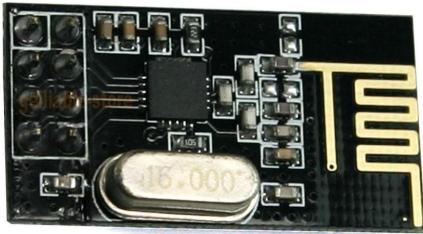
- Bootloaders, librerias, ide.
- Placas
- Perifericos
- Shields
- Software Libre
- Comunidad



Un Arduino para Cada Cosa

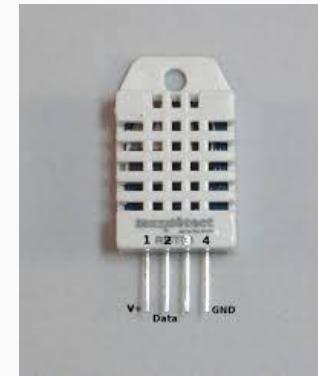


Perifericos



Perifericos

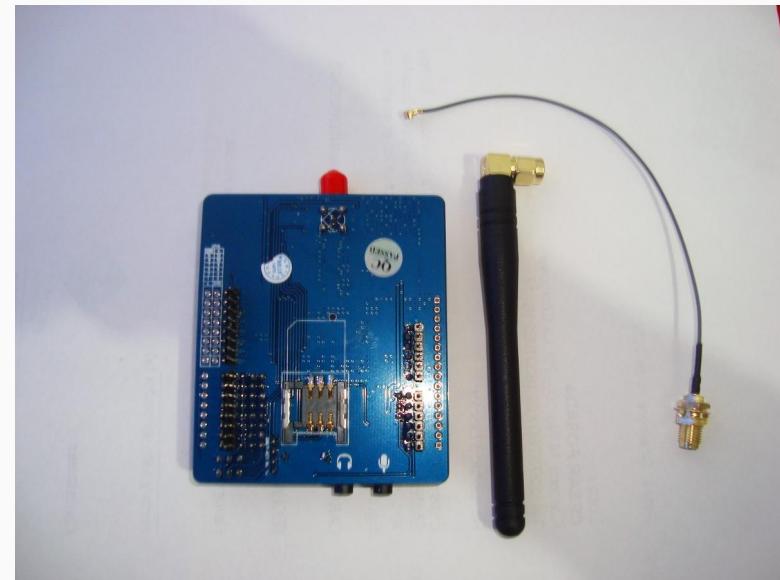
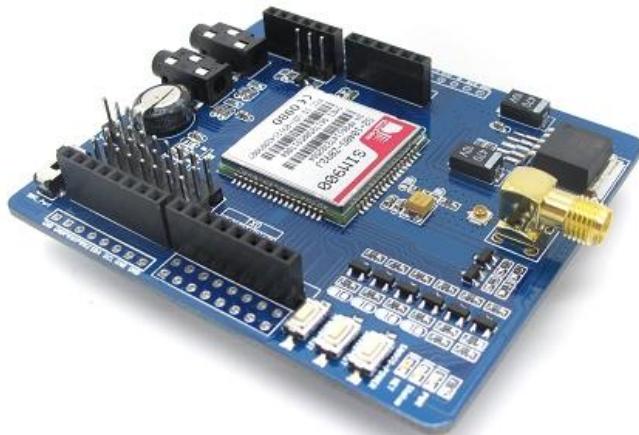
Sensor de temperatura y Humedad



Perifericos



SHIELDS



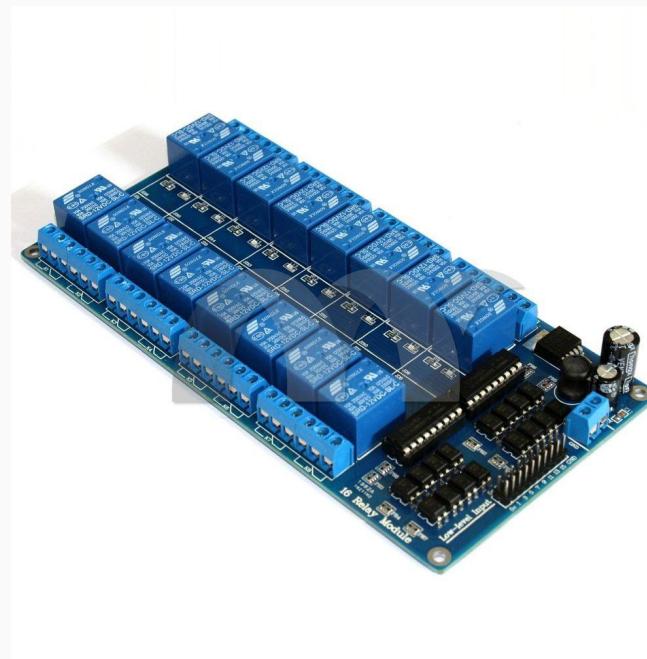
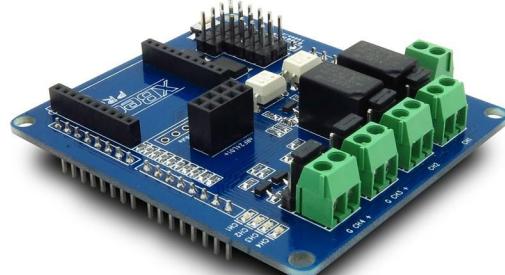
SHIELDS

Part



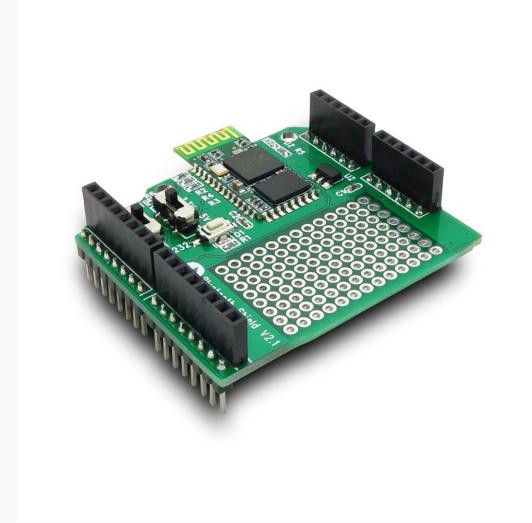
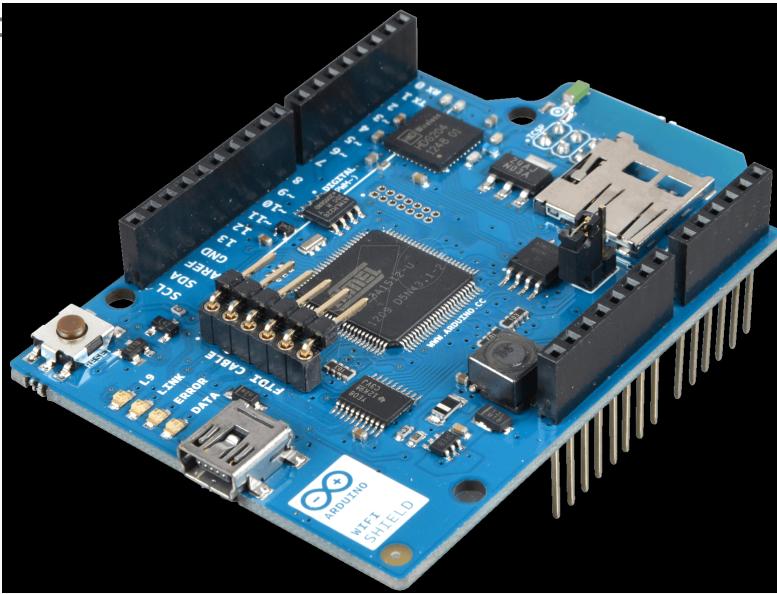
SHIELDS

R



SHIELDS

WIFI

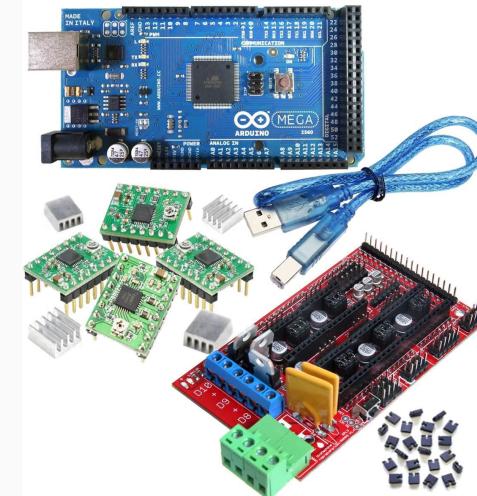


Frase poco celebre

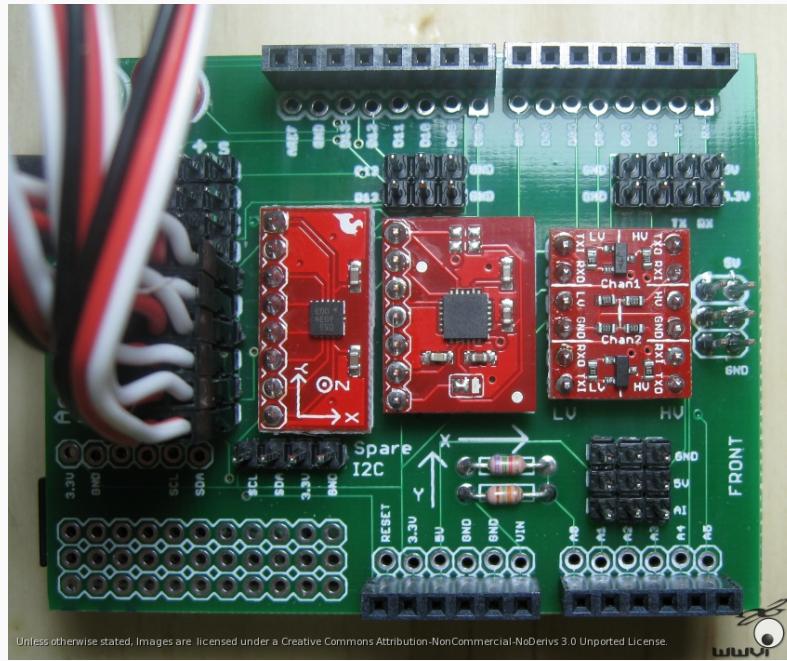
*“Para cosas serias no podes
usar arduino, es solo un
Juguete”*



Impresora 3D



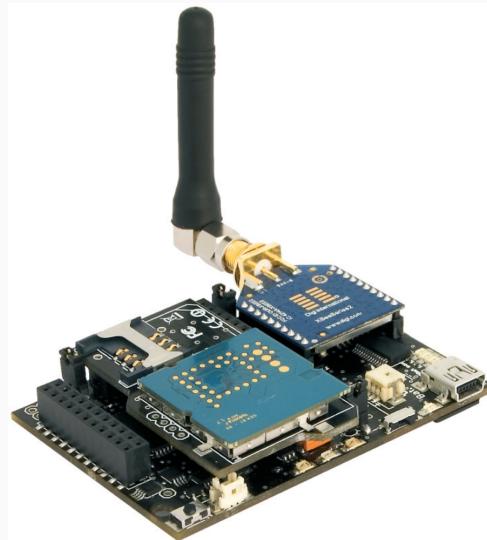
Quadcopter



Unless otherwise stated, Images are licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License.



Libelium



Mi experiencia



- 1º mes prototipo: comunicación Inalámbrica y GSM
- 2º mes equipo autónomo, consumo 100 veces menor



Nuestro primer proyecto

El lenguaje para arduino.

- Es una adaptación de C++ y avr-gcc.
- Posee muchas librerías propias integradas y otras que se pueden bajar.
- Hoy solo tenemos que aprender, Variables y Funciones.



Variables

Standard C++

char

byte

int

unsigned int

long

unsigned long

float

double

Objeto

String



Funciones

```
tipo_dato nombreDeLaFuncion(parametro/s)
```

```
{
```

```
    código;
```

```
    return(parametro_de_vuelta);
```

```
}
```



¿Como empiezo?

Software necesario:

- Arduino IDE

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

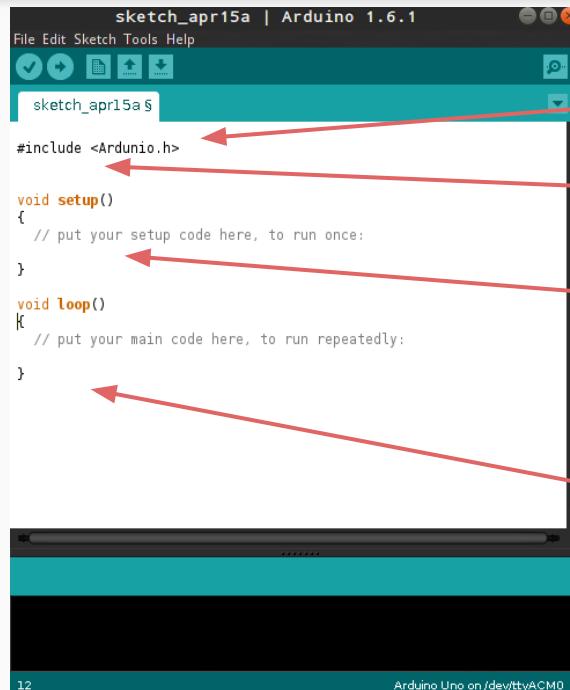


Código Arduino

- Proyecto = Sketch
- Se guarda como código .ino
- 2 Funciones obligatorias:
 - **setup()**
 - **loop()**



Estructura de código arduino



The screenshot shows the Arduino IDE interface with the title bar "sketch_apr15a | Arduino 1.6.1". The code editor contains the following Arduino sketch:

```
#include <Arduino.h>

void setup()
{
  // put your setup code here, to run once:
}

void loop()
{
  // put your main code here, to run repeatedly:
}
```

The status bar at the bottom indicates "Arduino Uno on /dev/ttyACM0".

Librerías

Variables y Funciones

Inicialización

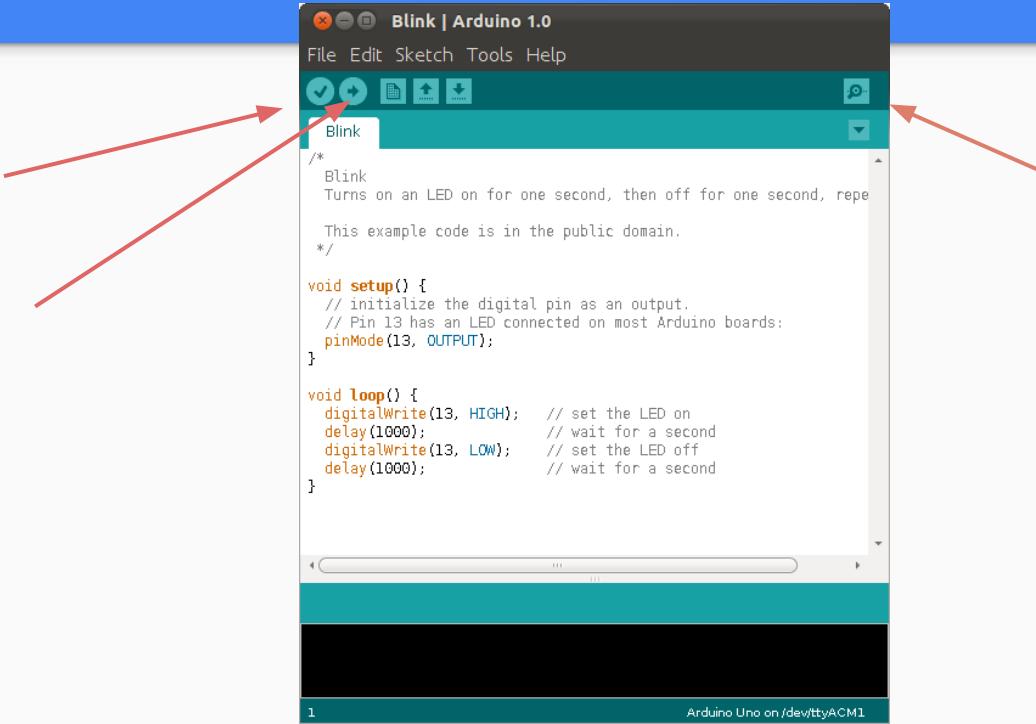
Ejecución



IDE

Compilar

Grabar



Serial



Descansemos 15 min



Comienza lo interesante



¿Cómo comienzo?

- Instalar IDE arduino.
- Y conectarlo al puerto USB.
- Verifiquemos en Herramientas/puerto si está conectado.



El serial - Nuestro Debugger

//Configuracion

Serial.begin(VelocidadSerial);

//Ejecución

Serial.print("Palabra entre comillas");

Serial.println("Palabra entre comillas");

Serial.println(Variable);



Hola Mundo!!!

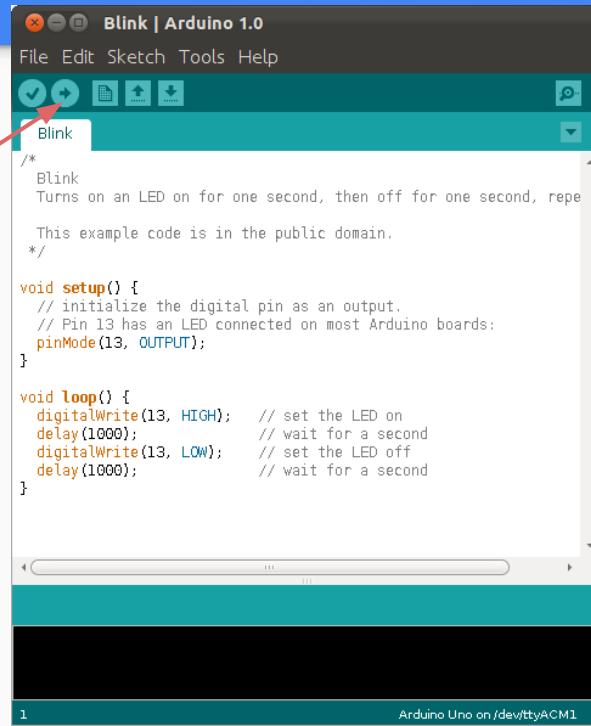
```
void setup()
{
    Serial.begin(9600); //Configuracion
    Serial.print("Hola mundo!!"); //Ejecución
}
```



Ahora

Compilar

Grabar



Serial



Delay y loop

- Loop es un bucle infinito.
- Se ejecuta a la velocidad del microC.
- Para que sea visible colocamos una espera.
- Función: **delay(ms);**
- 1000ms = 1 segundo
- Hagamos un “hola mundo” que muestre cada 2 segundos .



Recibiendo en el serial

- `Serial.available()`
- Verifica si hay caracteres para leer.
- Devuelve la cantidad de bytes disponibles.
- Se usa `if (Serial.available() > 0)`



Recibiendo en el serial

- `Serial.read()`
- Recibe el primer byte disponible
- Se usa `dato = Serial.read();`



Enviando y recibiendo valores

```
int dato_recibido = 0; // para  
almacenar la info recibida  
void setup()  
{  
    // iniciamos el serial  
    Serial.begin(115200);  
}
```

```
void loop() {  
    if (Serial.available() > 0) // verifico si recibo:  
    {  
        // leo el dato recibido:  
        dato_recibido = Serial.read();  
        // y lo imprimo  
        Serial.print("El dato recibido fue: ");  
        Serial.println(dato_recibido, DEC);  
    }  
}
```

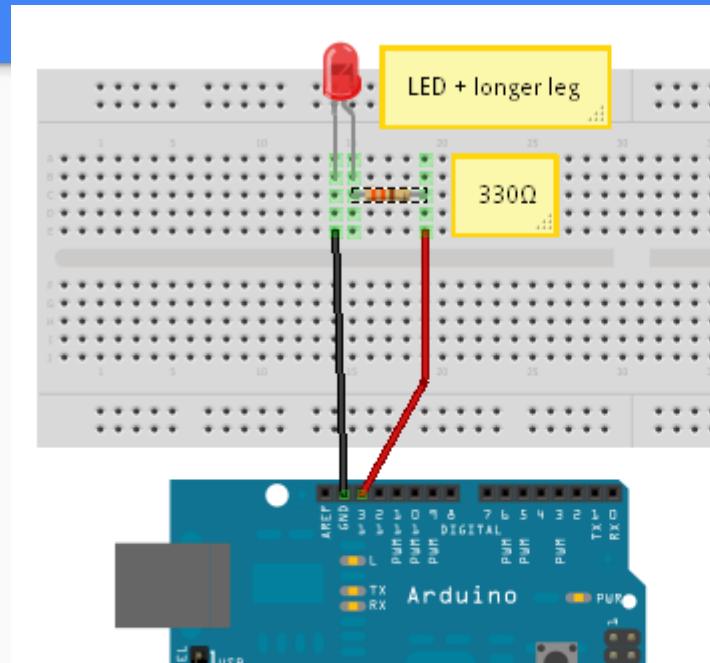
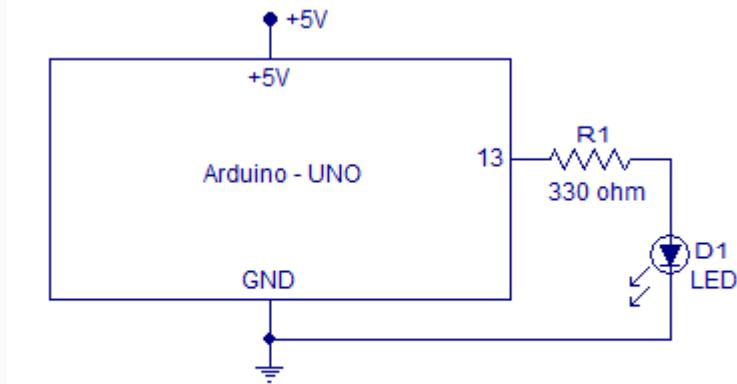


Conectando un Led

- Para un led debemos limitar el paso de la corriente.
- Un led consume entre 18 y 22 ma y Arduino funciona a 5v.
- Ley de ohm $R= V/I \rightarrow R=(5/0.018)=277.8$ ohm
- 2 valores comerciales 220 ohm o 330 ohm



Conectando un Led



Entrada y salida

```
//Configuracion  
pinMode(pin, INPUT/OUTPUT);  
  
//Ejecución  
digitalWrite(pin, HIGH/LOW);
```



Blink

```
void setup()
{
// Establece un pin digital como
salida.
// El Pin 13 tiene un LED
conectado a el.
pinMode(13, OUTPUT);
}
```

```
void loop()
{
digitalWrite(13, HIGH); // enciende led
delay(1000); // espera un segundo
digitalWrite(13, LOW); // apaga el LED
delay(1000); // espera un segundo
}
```



Función Random

- **randomSeed(analogRead(0));** toma un valor de ruido para generar una semilla aleatoria
- **random(max);** devuelve un número aleatorio entre 0 y max. Nosotros usaremos entre 0 y 1 (colocamos valor 2)
- LOW =0 HIGH=1



Jugando con 4 leds y random

```
int pines[]={10,11,12,13};  
void setup()  
{  
    int i=0;  
    for (i=0;i<4;i++)  
    {  
        pinMode (pines[i], OUTPUT)  
    }  
    randomSeed(analogRead(0));  
}
```

```
void loop()  
{  
    int i=0;  
    for ( i=0;i<4;i++)  
    {  
        digitalWrite( pines[i], random(2));  
    }  
    delay(1000);  
}
```



Consultas:



abdalajc@gmail.com



@toniabdala

