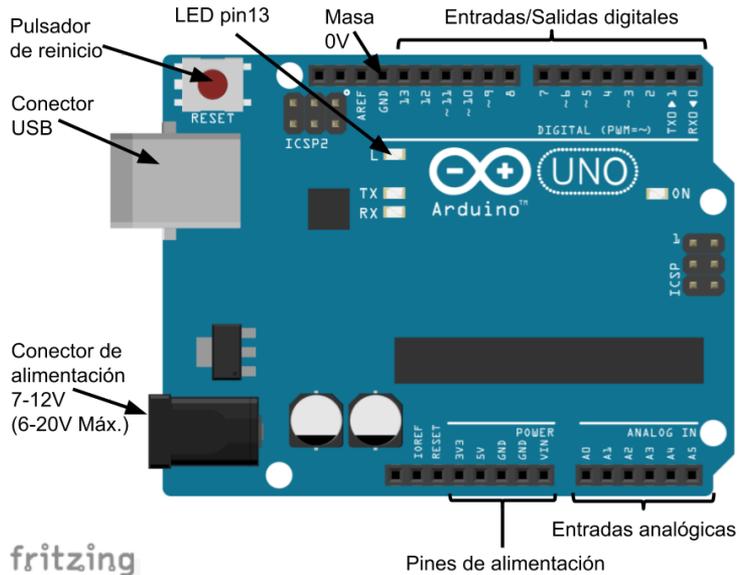


# PRONTUARIO DE ARDUINO

## COMPONENTES ELECTRÓNICOS BÁSICOS (HARDWARE)

### Tarjeta: Arduino UNO

- Microcontrolador ATMEGA328
- Reloj de 20 MHz
- Pulsador de Reset
- Entradas/Salidas digitales 14
- Entradas analógicas 6
- Salidas PWM 6 (Pin:3,5,6,9,10 y 11)
- Conexión USB a ordenador
- Alimentación de 6 a 12V o por USB.



**LED:** Rojo, Ámbar, Verde.

$I_{LED}$  entre 5 y 20 mA

$V_{LED}$  entre 1.3 y 1.7V

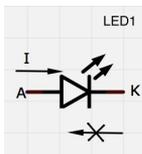
Resistencia de para 5V suele ser de 220, 330 o 470Ω.

Se puede compartir resistencia siempre que no supere la potencia máxima que admite.

Fórmula:

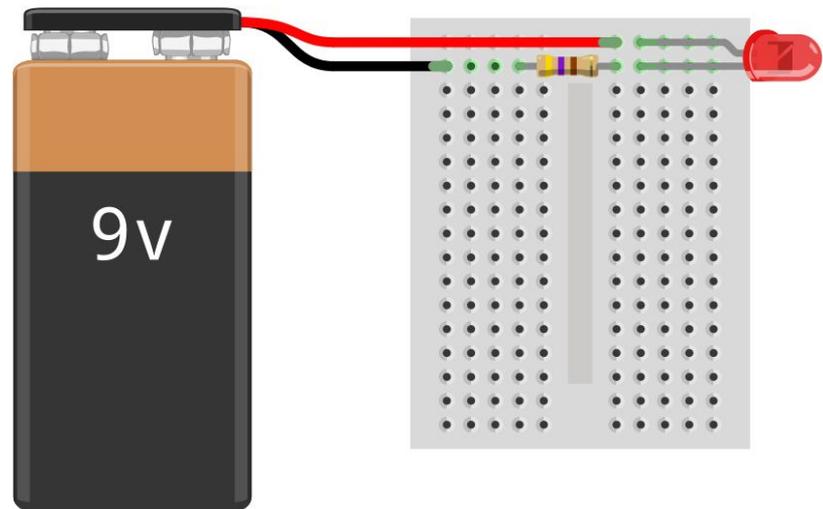
$$R_{m\acute{a}x} = \frac{V - V_{LED}}{I_{m\acute{i}n}} = \frac{5 - 1.3}{0.005} = 740\Omega$$

$$R_{m\acute{i}n} = \frac{V - V_{LED}}{I_{m\acute{a}x}} = \frac{5 - 1.3}{0.02} = 185\Omega$$



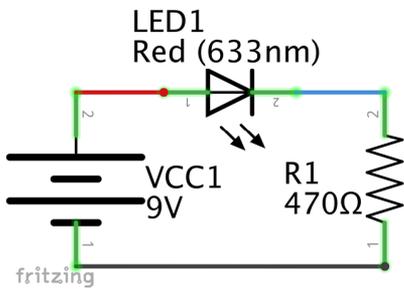
Conexionado:

Ejemplo de polarización de un LED con batería de 9V y resistencia de 470Ω

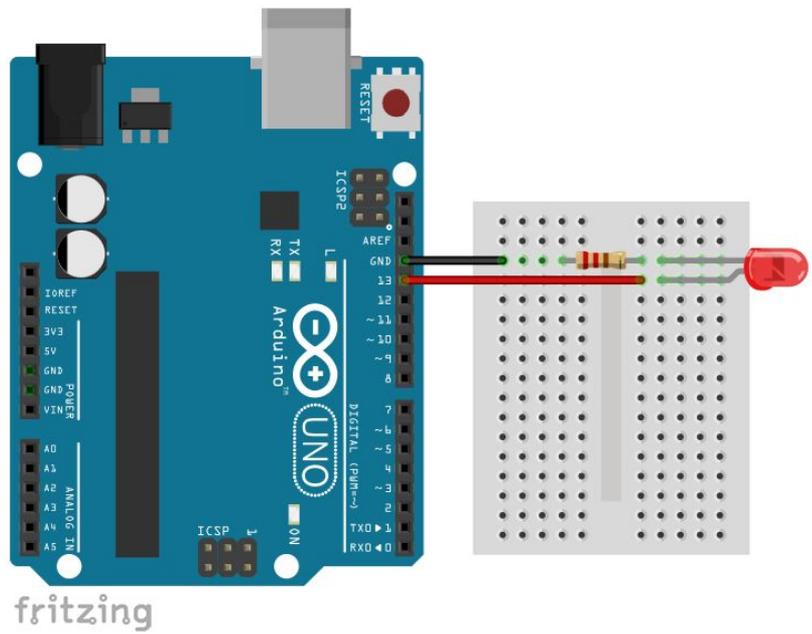


fritzing

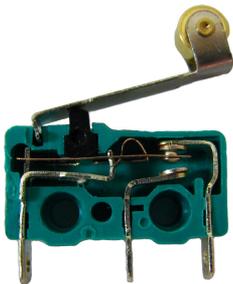
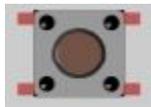
Ejemplo de LED rojo conectado el ánodo al pin13 con resistencia de 220Ω al cátodo y ésta a masa (GND). Se enciende cuando el Pin13 ofrece 5V (HIGH)



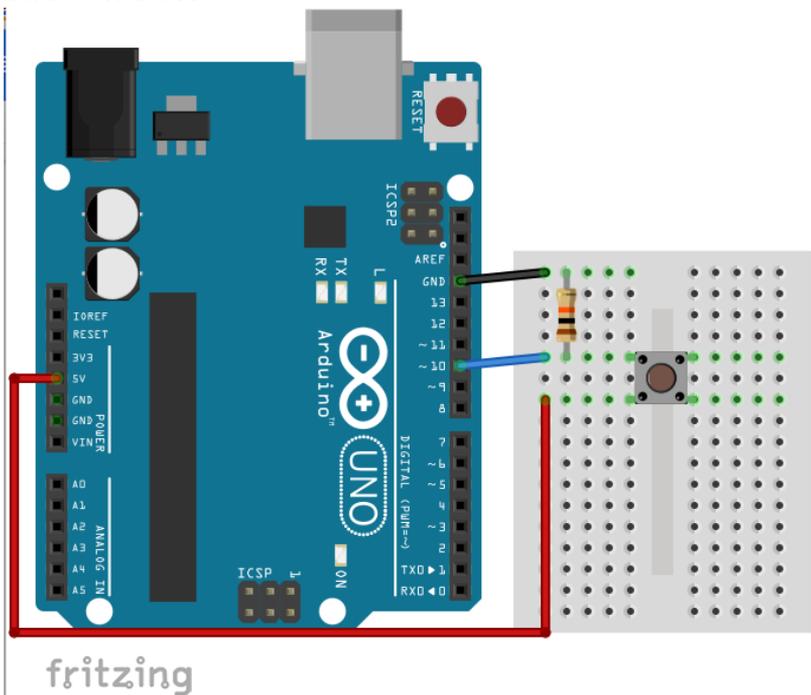
¿Colocación del diodo en el esquema????



Pulsador: Botón o Final de carrera



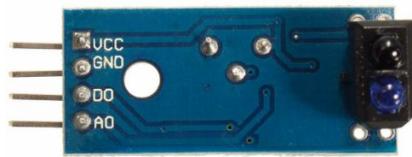
Conexionado: ejemplo de un pulsador que cuando es pulsado ofrece un "1" (5V) y un "0" (0V) sin pulsar al pin10 con ayuda de una resistencia de 10K a masa (GND) y el otro conector del botón a 5V.



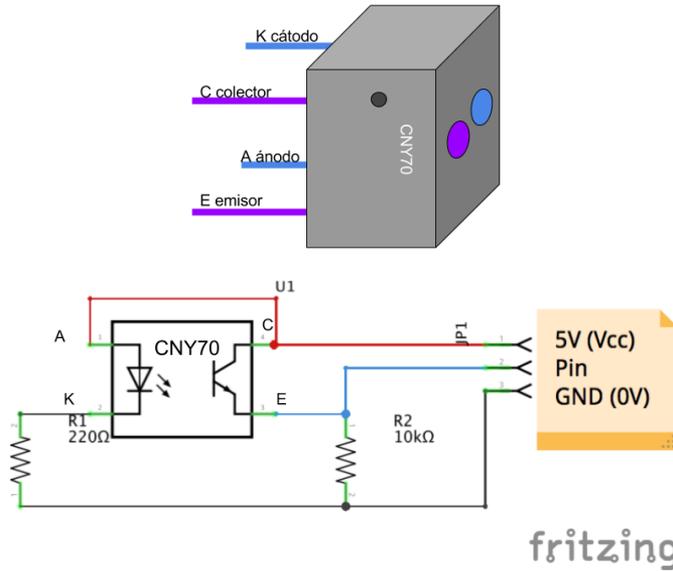
Sensor Infrarrojo (Emisor-Receptor) CNY70 (para montarlo tú)



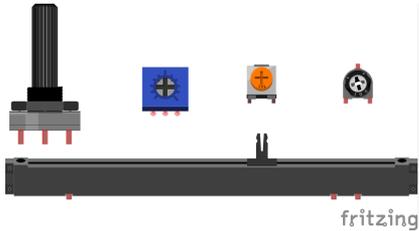
Módulo sensor infrarrojo de líneas



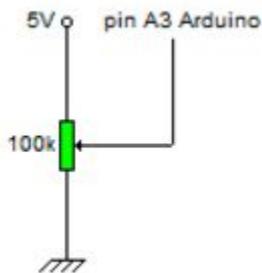
Conexionado:



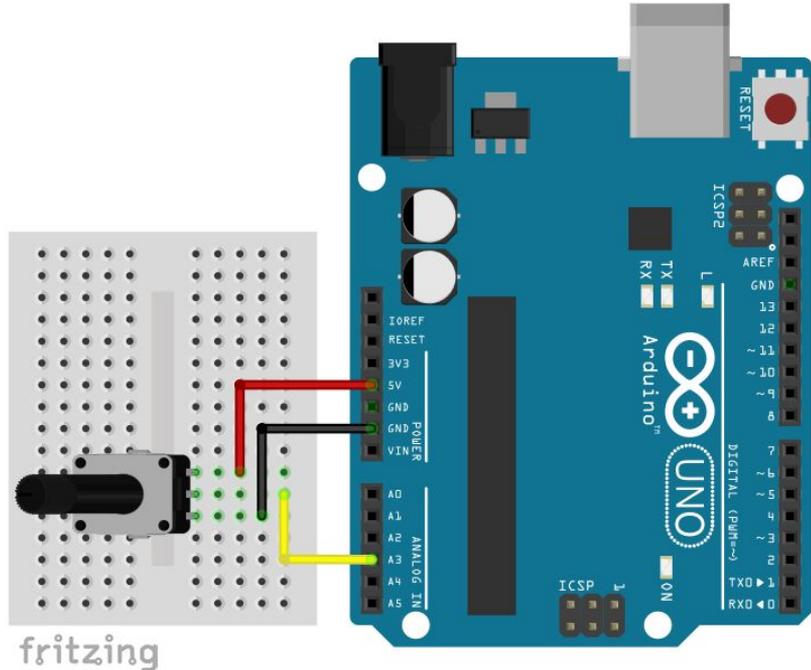
Potenciómetro



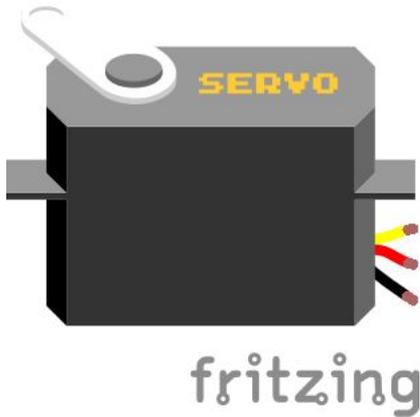
3



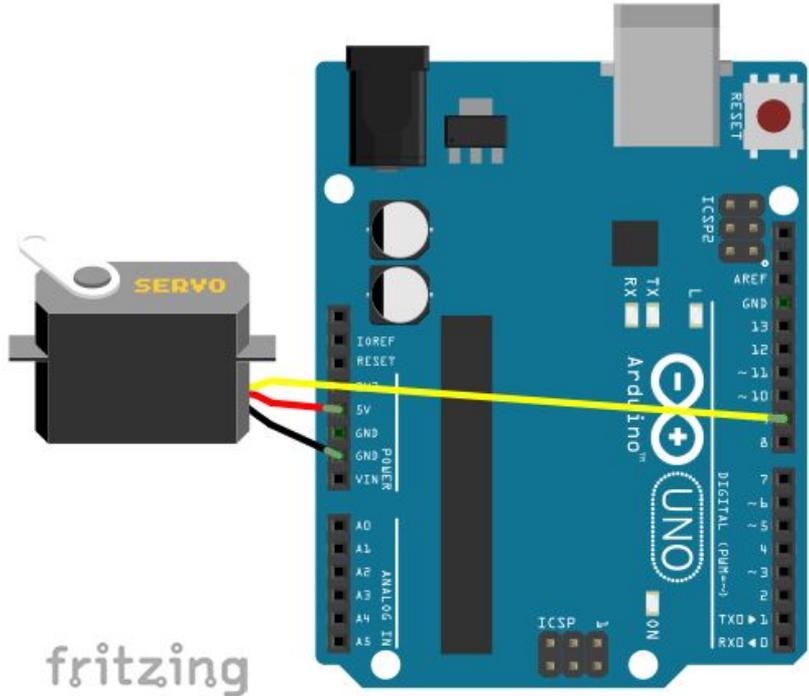
El potenciómetro da mucho juego, pero normalmente se emplea como sensor analógico como divisor de tensión. El valor que recibe la entrada A3 está entre 5 y 0V.



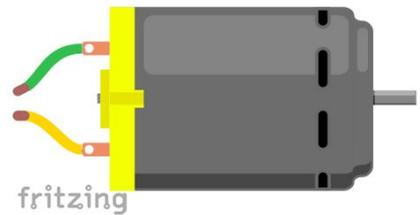
Servo (Motor) angular o de rotación continua



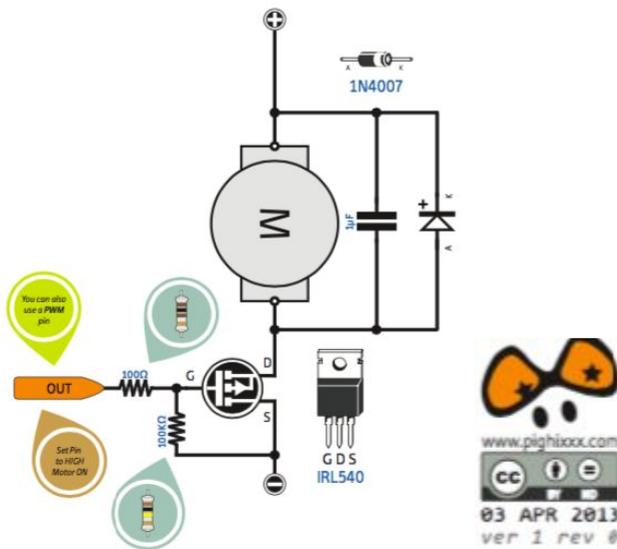
Conexión: En el ejemplo conectado a D9



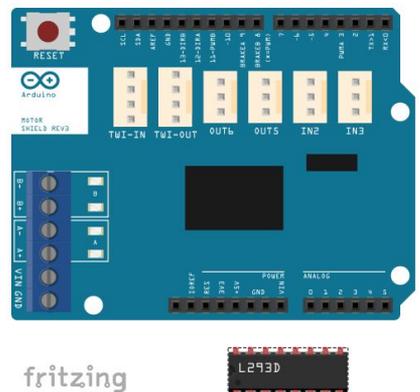
Motor de imán permanente



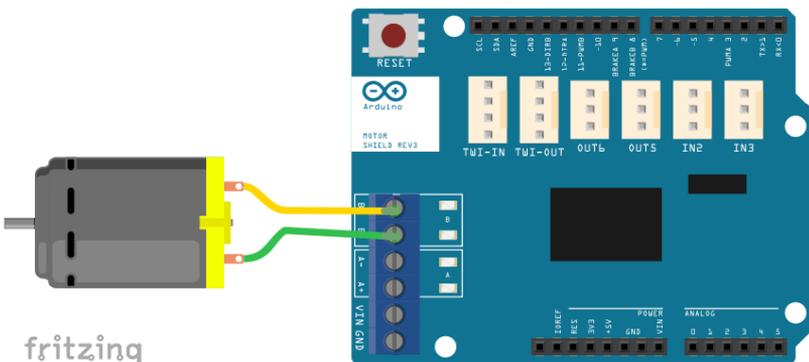
Conexión: con transistor



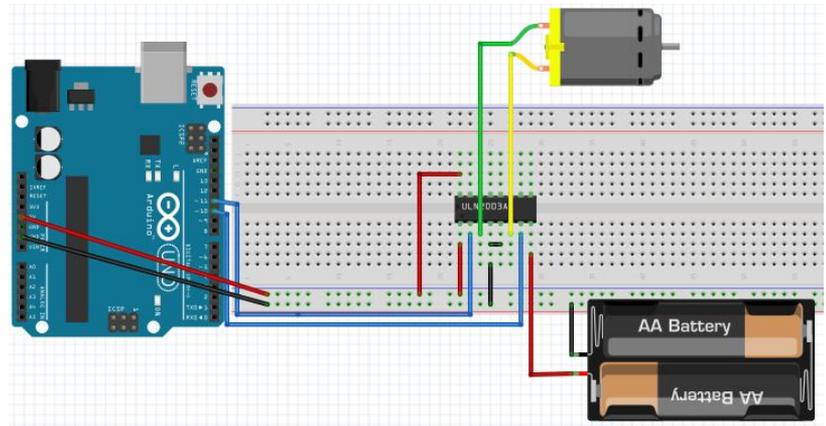
Driver Motor de continua



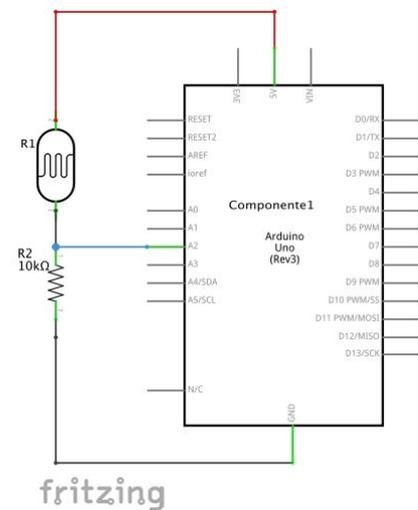
Permite la conexión de dos motores A y B.



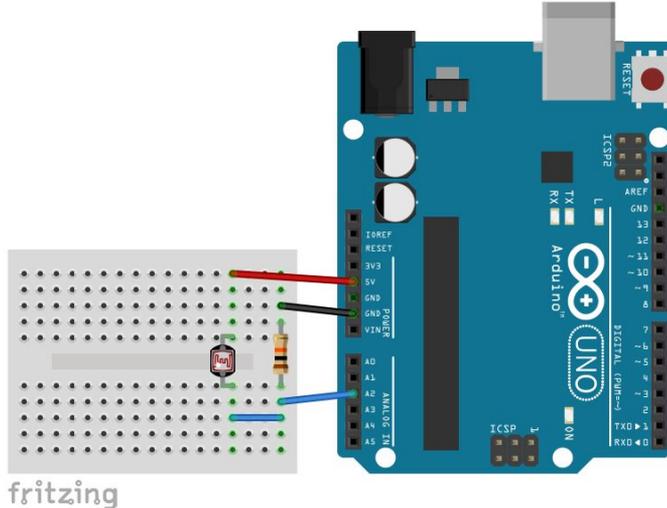
### Motor de corriente continua con integrado L293D



### LDR



Conexión: se trata de una resistencia que depende de la intensidad luminosa (oscuridad  $\rightarrow$  M $\Omega$  y máxima luz  $\rightarrow$   $\Omega$ ). Tiene un problema que comparte con los LED, admiten poca intensidad de corriente por su seno. Se suelen emplear como sensores para interruptores crepusculares (encienden una lámpara cuando no hay mucha luz). Para ello se hace un divisor de tensión con una resistencia de 10K $\Omega$ .

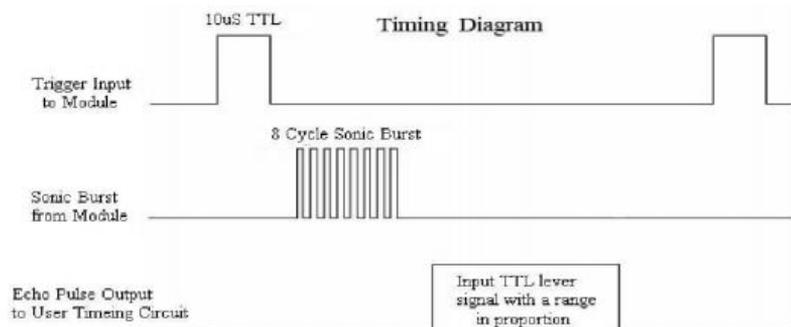


### Sensor de ultrasonidos HC-SR04



se trata de un dispositivo ultrasónico de bajo coste para determinar distancias cortas de 2cm a 4m con precisión de 3mm. Su determinación se fundamenta en el tiempo que tarda en volver una onda

Conexión:

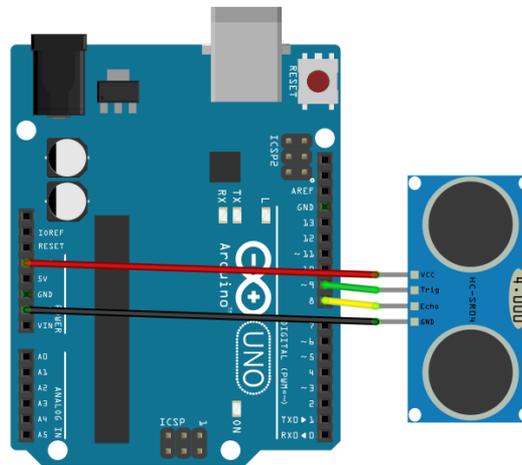


ultrasonica de 40kHz (ojo con los animales domésticos).  
Se envía un pulso de  $10\mu s$  → se cuenta el tiempo en volver → se convierte a centímetros.

La fórmula es

$$V = \frac{2 \cdot d}{t} = \frac{340m}{s} \cdot \frac{1s}{1000000\mu s} \cdot \frac{100cm}{1m} = \frac{2 \cdot d}{t}$$

$$d(cm) = \frac{0,34 \cdot t(\mu s)}{2} = 0,17 \cdot t(\mu s)$$



fritzing

Puedes emplear la librería correspondiente "Ultrasonic.h" o emplear la fórmula (esta última permite calcular velocidades si se solicita cada 60ms)

## Plantilla de Arduino

```
// Estos símbolos permiten escribir comentarios de una línea
// Declaración de variables globales (para todo el programa)
void setup() {
  // Este bloque sólo se ejecuta al cargar el programa
}
void loop() {
  // Este bloque se ejecuta repetidamente
  // Declaración de variables locales (de la función)
  // Llamadas a funciones
}
tipo mifuncion(argumentos) {
  // Este bloque sólo se ejecuta cuando es llamada
  // Tipo es el tipo de dato que se devuelve como las variables (void, int, ...)
}
```

## Tabla de funciones

Función	Descripción	Ejemplo
<b>Configuración de Entradas y Salidas</b>	En la función setup(), se definen el modo de trabajo de las patillas de la tarjeta con el comando pinMode(pin, Modo): <ul style="list-style-type: none"> <li>Pin -&gt; número de pin de la tarjeta</li> <li>Modo:               <ul style="list-style-type: none"> <li>Entrada -&gt; INPUT, 0</li> <li>Salida -&gt; OUTPUT, 1</li> </ul> </li> </ul>	<pre>void setup() {   // Pin 13 como salida   pinMode(13, OUTPUT);    // Pin 10 como entrada   pinMode(10, INPUT); }</pre>
digitalWrite(pin,valor);	<b>Escribe en la salida</b> del pin el valor digital: <ul style="list-style-type: none"> <li>pin: número de pin de la tarjeta</li> <li>valor:               <ul style="list-style-type: none"> <li>HIGH , 1-&gt; 5V</li> <li>LOW, 0 -&gt; 0V</li> </ul> </li> </ul>	<pre>// Envía "1" (5V) a la patilla 13 digitalWrite(13, HIGH);  //Envía "0" (0V) a la patilla 11 digitalWrite(11, LOW);</pre>

Función	Descripción	Ejemplo
digitalRead(pin);	<p><b>Lee en la entrada</b> del pin el valor:</p> <ul style="list-style-type: none"> <li>● pin: número de patilla de la tarjeta</li> <li>● valor: este puede ser: <ul style="list-style-type: none"> <li>○ 0, LOW -&gt; 0V en el pin</li> <li>○ 1, HIGH-&gt; 5V en el pin</li> </ul> </li> </ul>	<pre>//Leo la entrada 10&gt; la guardo en y y=digitalRead(10);  // Si hay "0" en pin10 -&gt; espero 1s if (digitalRead(10)==0) {   delay(1000); }</pre>
delay(miliseundos); delayMicroseconds(microseg)	<p><b>Espera o paraliza el programa</b> los:</p> <ul style="list-style-type: none"> <li>● miliseundos</li> <li>● microsegundos</li> </ul>	<pre>// Espero 2 segundos =2000ms delay(2000);</pre>
analogWrite(pin,valor);	<p><b>Manda en el pin de salida un valor:</b></p> <ul style="list-style-type: none"> <li>● Definir el pin como SALIDA <ul style="list-style-type: none"> <li>○ Sólo pin 3,5,6,9,10 y 11</li> <li>○ Son las que tienen un símbolo de alterna</li> </ul> </li> <li>● valor: ofrece una señal periódica de pulso variable PWM. 1KHz <ul style="list-style-type: none"> <li>○ Led (255&gt;brillo intenso) y (0&gt;no brilla)</li> <li>○ Motor-&gt;variador de velocidad (255 vel.max.)</li> </ul> </li> </ul>	<pre>void setup() {   pinMode(3,OUTPUT) } void loop(){   // Luce poco el led   analogWrite(3, 125);   delay(1000); //espera 1s   //luce mucho el led   analogWrite(3,255);   delay(1000); //espera 1s }</pre>
analogRead(analogPin);	<p>Lee el valor de la patilla analógica:</p> <ul style="list-style-type: none"> <li>● No hay que definirla como entrada</li> <li>● Sólo se pueden utilizar: <ul style="list-style-type: none"> <li>○ A0, A1, A2, A3, A4 y A5</li> </ul> </li> <li>● El valor hay que guardarlo en una variable entera y oscila entre (0-1024)</li> </ul>	<pre>//zona de variables globales int valor=0; ... //leo el valor de la entrada A0 valor=analogRead(A0);</pre>
tone(pin,frecuencia,duracion) noTone(pin);	<p><b>Emite una onda</b> cuadrada, generalmente <b>para producir un sonido:</b></p> <ul style="list-style-type: none"> <li>● Pin: solo salidas 3,5,6,9,10 y 11</li> <li>● frecuencia (sonido) en microseg.</li> <li>● duración en miliseundos.</li> </ul>	<pre>// Emito Do en la patilla 3 corchea // Zumbador +R=100Ohm serie tone(3,261,500); // paro el sonido noTone(3);</pre>
<b>Comunicación USB</b>	<p>Serial.setTimeout(miliseundos)</p> <ul style="list-style-type: none"> <li>● Tiempo que se espera a la escucha del USB.</li> <li>● Necesario para readBytesUntil()</li> </ul>	<pre>void setup() {   Serial.begin(9600);   // Espera 10s   Serial.setTimeout(10000); }</pre>
Serial.begin(vel_trans);	<p>Configura la transmisión USB de la tarjeta con el Ordenador.</p> <ul style="list-style-type: none"> <li>● vel_trans -&gt; velocidad de la transmisión en baudios (bits/seg).</li> </ul>	<pre>void setup() {   // transmisión a 9600 baudios   Serial.begin(9600);   ... }</pre>
Serial.print(valor); Serial.println(valor); Serial.write(variable); Serial.write(buffer,length);	<p><b>Manda al ordenador por USB</b> el contenido de la variable "valor".</p> <ul style="list-style-type: none"> <li>● Tiene que haberse configurado la comunicación (ver anterior punto)</li> <li>● println() es como print() pero añade a "valor" un final de línea (EOL = End Of Line)</li> </ul> <p>Donde variable puede ser:</p> <ul style="list-style-type: none"> <li>● Número</li> <li>● string: cadena de caracteres</li> <li>● buffer: vector de valores</li> <li>● length: longitud del vector</li> </ul>	<pre>// Mando el contenido de jj y EOL Serial.println(jj);  // Mando el contenido de jj Serial.print(jj);</pre>

Función	Descripción	Ejemplo
<pre>Serial.read(); Serial.parseInt(); Serial.readBytes(buffer,length) ;  Serial.readBytesUntil(caracter, buffer,length);</pre>	<p><b>Escucha al ordenador por USB:</b></p> <ul style="list-style-type: none"> <li>• read(): lee un byte</li> <li>• buffer: vector de bytes</li> <li>• length: longitud del vector</li> <li>• caracter: letra que sirve para parar la escucha.</li> <li>• parseInt(): devuelve el nº entero.</li> </ul>	<pre>// Lee un byte del PC y guarda en y y=Serial.read();</pre>
<p><b>DECISIONES Y CONDICIONES</b></p>	<p>En determinadas condiciones hay que hacer o dejar de hacer el ciclo normal de repeticiones</p>	
<pre>if(condicion) { // si se cumple } else { // si no se cumple }</pre>	<p>No es imprescindible "else"</p> <ul style="list-style-type: none"> <li>• <b>Si se cumple</b> la condición, entonces se hace lo que hay entre las llaves de después de if</li> <li>• <b>si no se cumple</b>, lo que se ponga después de else</li> </ul> <p>Los operadores para la if:</p> <ul style="list-style-type: none"> <li>• == igual, != distinto</li> <li>• &gt; mayor, &lt; menor</li> <li>• &gt;= mayor o igual, &lt;= menor o igual</li> <li>• &amp;&amp; y,    o</li> </ul>	<pre>// Si i=3 =&gt; enciende LED13 if(i==3) { digitalWrite(13,HIGH); }  // Si j&gt;4 =&gt; enciende LED13 // si no =&gt; apaga LED13 if (j&gt;4) { digitalWrite(13,HIGH); } else { digitalWrite(13,LOW); }</pre>
<pre>for( cond_ini; cond_d; modif) { // repeticiones si cumple }</pre>	<p><b>Desde</b> la condición inicial "cond_ini", <b>hasta</b> condición "cond_d", se realiza lo que hay entre las llaves modificando la variable según "modif". Operadores válidos: =,&gt;,&lt;,&gt;=,&lt;=</p>	<pre>// Durante 10 segundos parpadea //el LED13 for (i=1; i&lt;=10;i=i+1) { digitalWrite(13,HIGH); delay(500); //espero 1s digitalWrite(13,LOW); delay(500); }</pre>
<pre>while(condición) { // si se cumple }</pre>	<p><b>Mientras</b> se cumpla la condición, se repite lo que haya entre las llaves</p>	<pre>// Lee 10 valores cada 1ms int i[]; for (i&lt;11) { delayMicroseconds(998); lee[i]=digitalRead(1); i=i+1; }</pre>
<pre>switch(variable) { case valor1: // Hacer si cierto break; case valor2: // Hacer si cierto break; ... case valorn: // Hacer si cierto break; default: // Hacer si no cierto ninguna break; }</pre>	<p>Para la "variable" se hacen las sentencias según el valor, si no coincide ninguna, se realiza lo que hay en default</p>	<pre>switch(barrido) { case '1': lee=digitalRead(1); break; case '2': lee=digitalRead(2); break; case '3': lee=digitalRead(3); break; default: lee=digitalRead(10); break; }</pre>
<p><b>INTERRUPCIONES</b></p>	<p>Permiten interrumpir el normal desarrollo para ejecutar la función correspondiente.</p>	
<pre>attachInterrupt(i,f,m);  detachInterrupt(i);</pre>	<p><b>Asigna una función a una señal externa de interrupción:</b></p> <ul style="list-style-type: none"> <li>• i: interrupción 0-&gt;pin1 y 1-&gt;pin2</li> <li>• f: nombre de la función a ejecutar</li> </ul>	<pre>volatile int state=LOW; //cambiara valor void setup() { pinMode(13, OUTPUT); //LED attachInterrupt(0, cambia(), CHANGE);</pre>

	<ul style="list-style-type: none"> <li>• m: modo de detectar la interrupción             <ul style="list-style-type: none"> <li>○ LOW -&gt; pin a 0</li> <li>○ CHANGE -&gt; pin cambia</li> <li>○ RISING -&gt; pin pasa 0-&gt;1</li> <li>○ FALLING-&gt; pin pasa 1-&gt;0</li> </ul> </li> <li>• detachInterrupt: elimina asignación</li> </ul>	<pre>//Interrupcion en pin1 } void loop() { digitalWrite(13, state); } void cambia() {   state = !state; // si es 1 -&gt; 0 o 0 -&gt; 1 }</pre>
--	--	---

## Tabla de frecuencias y notas (tone)

Octava	Do (C)	Do#	Re (D)	Re#	Mi (E)	Fa (F)	Fa#	Sol (G)	Sol#	La (A)	La#	Si (B)
0												31
1	33	35	37	39	41	44	46	49	52	55	58	62
2	65	69	73	78	82	87	93	98	104	110	117	123
3	131	139	147	156	165	175	185	196	208	220	233	247
4	262	277	294	311	330	349	370	392	415	440	466	494
5	523	554	587	622	659	698	740	784	831	880	932	988
6	1047	1109	1175	1245	1319	1397	1480	1568	1661	1760	1865	1976
7	2093	2217	2349	2489	2637	2794	2960	3136	3322	3520	3729	3951
8	4186	4435	4699	4978								

Imágenes y esquemas desarrollados con Fritzing bajo licencias GPLv3 y Creative Commons BY-SA en <http://fritzing.org/home/>

Autor: Fco. Javier Arteaga

Publicado por: Centro Regional de Formación del Profesorado de Castilla-La Mancha

<http://centroformacionprofesorado.castillalamancha.es/comunidad/crfp>

Bajo licencia Creative Commons 4.0 con reconocimiento – NoComercial – CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.