Introducción a la electrónica con JavaScript

Iniciación al desarrollo en Arduino con JavaScript y Node.js

Mate Marschalko

Iniciación a la electrónica con Javascript
por **Mate Marschalko**

Publicado en 2015 por **Web on Devices**,
Londres, Reino Unido.

Corrección de textos
y revisión por **Nicholas Headlong** y **Pedro Carmona**.
Revision técnica por **James Miller**.
Revisión de ingeniería eléctrica por **Gabor Szoke.**

Escrito por

Mate Marschalko

Tecnólogo Creativo Senior
y Desarrollador Web

Mate es un creativo tecnólogo y desarrollador web que trabaja actualmente para las más relevantes agencias creativas de Londres, Reino Unido. Trabaja desde hace 10 años y siente pasión por el desarrollo web. Le encanta explorar nuevas tecnologías y dispositivos para posteriormente programarlas con JavaScript. Entre sus prototipos siempre tiene un nuevo juguete construido con Arduino, Raspberry PI u otras herramientas de desarrollo.

Revisión técnica por

James Miller

Líder Tecnológico y
Tecnólogo Creativo

James es un líder tecnológico freelance y tecnólogo creativo con más de 10 años experiencia en la industria. Es un colaborador habitual de las revistas Net Magazine y Smashing Magazine. Tiene pasión por las aplicaciones híbridas, construcción de su propio hardware y por su amado equipo de fútbol - Luton Town.

Web on Devices

Integrando electrónica con Javascript y otras tecnologías Web

www.webondevices.com

TABLA DE CONTENIDOS

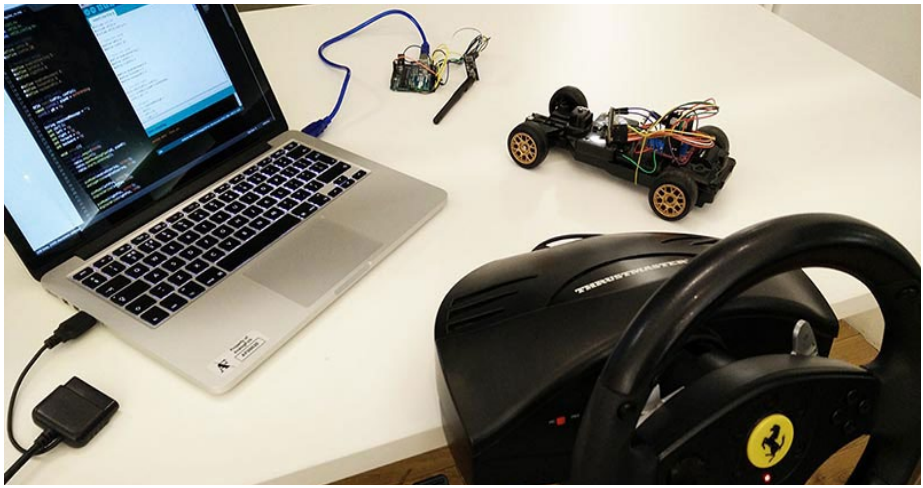
INTRODUCCIÓN

En los últimos años, JavaScript ha evolucionado más allá de ser simplemente una herramienta para la manipulación del DOM y ha comenzado a expandir su influencia más allá de los reinos de la ventana del navegador. Ahora vemos JavaScript apareciendo en los sistemas operativos (Chrome OS, Firefox OS), en el servidor (Node.js), e incluso en pequeños chips micro-controladores. También ha comenzado a convertirse en un ámbito común donde ver desarrolladores creando prototipos de hardware, o juguetes electrónicos, o el uso de JavaScript, en las plataformas de creación de prototipos como el Arduino. Estas plataformas y placas de desarrollo hicieron el trabajo con la electrónica posibles para aquellos sin experiencia en ingeniería eléctrica.

El Internet de las cosas, una red simple de dispositivos conectados, están allanando el camino a nuevas formas de comercializar y publicitar productos

a través de las experiencias físicas, además de ser una gran inversión financiera para los desarrolladores para cambiar sus habilidades en la construcción de aplicaciones físicas. Incluso hoy en día, las agencias de visión de futuro han comenzado a contratar tecnólogos creativos Front-end.

Estos desarrolladores no sólo son competentes creando sitios web, también son capaces de construir tiendas interactivas, experiencias de realidad virtual o de otros prototipos de hardware interactivo. Ellos no sólo ganan mucho más en comparación con los desarrolladores de aplicaciones para usuario regulares, sino que se ponen a experimentar con todos los últimos gadgets y juguetes en su trabajo del día a día!



Jugar con un coche a control remoto de Arduino

Este breve libro le ayudará a dar el primer paso en el mundo de la Creative Technology(tecnología creativa), y empezar a construir prototipos electrónicos con la placa de desarrollo Arduino UNO y JavaScript. Después de la introducción, usted aprenderá acerca de los conceptos básicos de la electricidad, de UNO y de sus componentes, el control de una luz LED y la lectura de los sensores de luz y temperatura. En el proceso, aprenderás cómo ejecutar código JavaScript en el servidor con Node. js, construir circuitos simples, y descubrir la librería Johnny-Five JavaScript para controlar y comunicarse con el Arduino UNO y sus componentes.

Arduino conectado a un Macintosh

RESTRICCIONES DEL NAVEGADOR

Si comparamos con las aplicaciones de escritorio y móviles, la ejecución de JavaScript en el navegador es bastante limitada. El acceso a los puertos de comunicación, componentes de hardware y el sistema de archivos está restringido. Este nivel de seguridad es necesario en Internet porque no todos los sitios web que visitamos son de confianza, por lo tanto, es necesario asegurarse de que el equipo no pueda ser infiltrado por algún código malicioso.

Una forma de evitar los bloqueos de seguridad es ejecutar la aplicación en un servidor, donde se puede transmitir todos los datos necesarios para el front-end. El servidor puede confiar en el código que se ejecuta en sí mismo y permitir el acceso a sus recursos de hardware. Hay muchas maneras de escribir aplicaciones de servidor, por ejemplo, correr PHP en un servidor Apache en Linux con el apoyo de una base de datos (MySQL, PostgreSQL, etc.), a veces conocido como LAMP. En

lugar de esta configuración JavaScript también se puede utilizar en nuestro propio equipo como un servidor.

La comunicación con el servidor tradicionalmente significaba el envío de solicitudes a través de HTTP y esperar a que el servidor responda. Alrededor de 2005, AJAX apareció haciendo de este proceso un poco más dinámico. AJAX es todavía usado para conectar con el servidor para cargar nuevos datos, pero esta vez sin la necesidad de recargar la página. Esto permitió un mayor control como la conexión y la respuesta que pueden ser gestionados mediante JavaScript.

Usar WebSockets ofrece un enfoque más eficiente para la recuperación de datos. El protocolo hace que sea posible abrir una sesión de comunicación de dos vías interactivo entre el navegador del usuario y un servidor. Con esta API (Application Protocol Interface) puede enviar mensajes a un servidor y recibir respuestas por eventos sin tener que sondear una respuesta.

El siguiente paso es construir el servidor que accede al puerto USB y lo cual pueda usar nuestra Arduino conectado a la misma, una tarea ideal para Node.js.

JAVASCRIPT EN EL SERVIDOR

Node.js es construido encima del motor V8 de JavaScript de Google que puede funcionar de forma independiente desde el navegador y en el servidor.

Drone AR conectado a Node.js

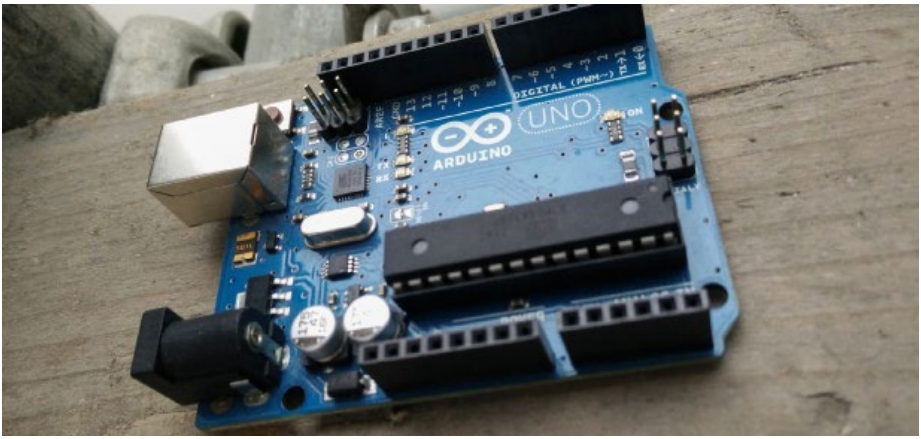
El uso de JavaScript y Node.js para el hacking de la electrónica no es nada nuevo. La comunidad ya lo está usando para alimentar proyectos exitosos como nodebots.io, Firmata, Cylon o Johnny-Five. El ecosistema Node.js es aún más fuerte, con más de cien

mil paquetes disponibles a través del gestor de paquetes node (www.npmjs.org). En términos de velocidad el rendimiento de JavaScript también ha mejorado de manera espectacular en la última década lo cual hace que este aspecto ya no sea un problema.

El evento impulsado, propiedades asíncronas también son excelentes para trabajar con la electrónica y sensores, permitiendo a las aplicaciones que se comunican a través de NFC, RFID o Bluetooth, todos los cuales son actualmente imposibles de usar en el navegador. Usando Node.js abre nuevas posibilidades para el desarrollo de aplicaciones utilizando dispositivos como el Xbox Kinect, Leap Motion, controladores midi, drones o accesorios para el hogar inteligentes como el termostato Nest o la bombilla Philips HUE.

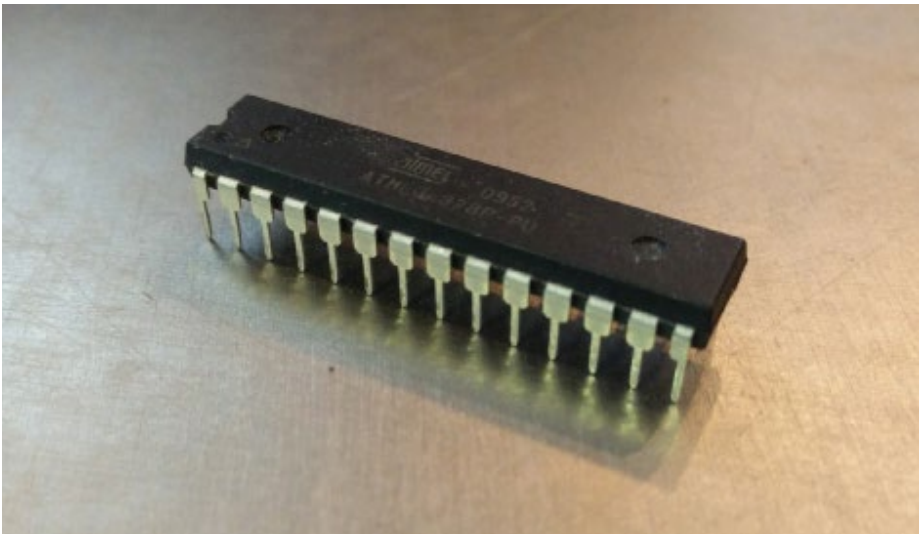
PLACAS DE DESARROLLO

La revolución para las placas de desarrollo comenzó alrededor de 2005, cuando la primera Arduino fue puesto en libertad. El proyecto Arduino fue iniciado por algunos profesores universitarios italianos tratando de hacer más fácil para los estudiantes el trabajar con los componentes electrónicos y para la programación de chips microcontroladores. Arduino sigue siendo, por mucho, la plataforma más popular hoy en día.



Placa de desarrollo Arduino UNO

El microcontrolador es el cerebro de la placa que controla y gestiona todo. Es de bajo coste, baja potencia, de bajo rendimiento y capaz de ejecutar una sola aplicación a la vez. El microcontrolador sólo es responsable de la gestión de entrada de bajo nivel y las señales eléctricas de salida, y la realización de algunos cálculos básicos con ellos. Este chip es capaz de conmutar los componentes eléctricos encendido y apagado, la medición de electricidad a partir de sensores, y la interconexión con otros componentes.

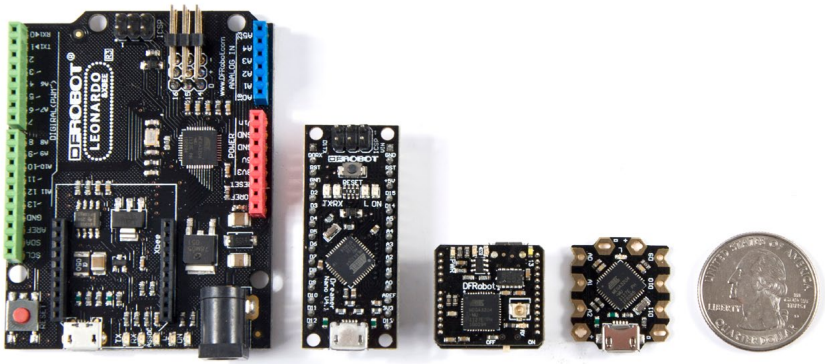


Microcontrolador de potencia de Arduino UNO

Estos chips pueden trabajar de forma independiente, sin embargo, las placas de

desarrollo pueden agilizar el proceso de desarrollo. Ayudan a potenciar, comunican, e interferir con el microcontrolador. La adición de un puerto USB para cargar el código a UNO, y un conector para alimentar el sistema a partir de una batería de 9V, son a la vez útiles.

Actualmente, hay una abrumadora cantidad de placas disponibles en el mercado para elegir. La mayoría de estos son compatibles con Arduino, así que ¿cómo son todos diferentes unos de otros?

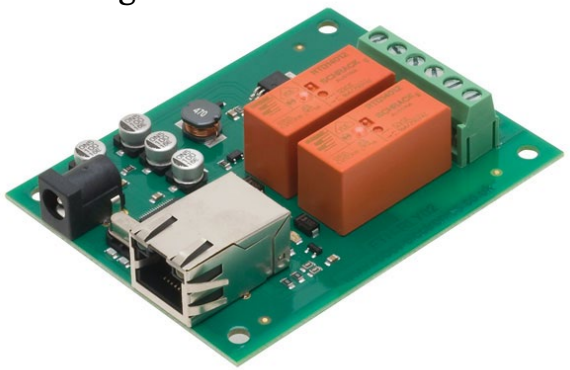


Placas compatibles con Arduino de diferentes tamaños, de DFRObot

Como punto de partida se diferencian mucho en tamaño. Uno de las placas más pequeñas disponibles son el DFRobot Beetle y la Tinyduino, que son ideales para proyectos “vestibles”(insertados en la ropa) o aviones

no tripulados (drones) donde el tamaño o el peso son factores importantes. Con un tamaño más pequeño que tiene una característica simplificada también, y menos componentes, lo que significa menor consumo de energía, por lo que son perfectos para prototipos con baterías integradas.

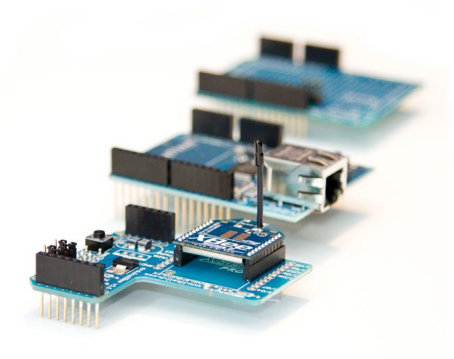
Otras tablas permiten funciones como Wi-Fi, Ethernet o conexión bluetooth, mientras que otros pueden actuar como un ratón, teclado u otro controlador USB cuando se conecta a su ordenador, también hay tablas diseñadas para la construcción de la domótica y seguridad en el hogar.



Placa de desarrollo para la automatización del hogar con dos transmisores para apagar y encender la red eléctrica

EL ARDUINO UNO

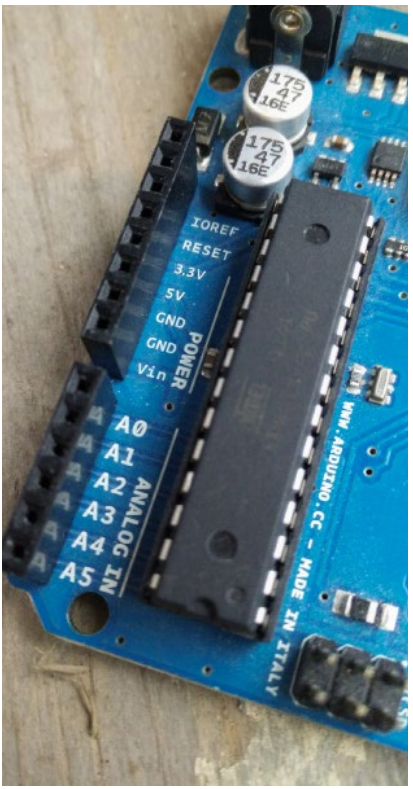
Estas placas son a la vez útiles y asequibles, pero la Arduino UNO sigue siendo la número uno, la placa más popular para empezar a desarrollar. Una de las razones podría ser que la estructura de los pines es limpia y amplia, por lo que es fácil para principiantes. Las características adicionales enumeradas anteriormente, como conexión inalámbrica, son posibles con los proyectos UNO, aunque éstos requerirán la adición de módulos o componentes externos.



Componentes externos para Arduino UNO

Anatomía de Arduino UNO

El componente más importante de la Arduino UNO es el chip micro-controlador ATmega328. Este chip gestiona los pines de entrada y salida (GPIO) de la UNO que se expone a través de los pines de cabecera a lo largo del borde superior e inferior de la placa. En resumen, los pines de entrada (ANALOG IN) están diseñados para medir la electricidad procedente de sensores como la luz, el sonido o la temperatura. Los pines de salida (DIGITAL) en el otro lado pueden cambiar la electricidad de manera intermitente para las luces, motores y otros componentes electrónicos. Tanto a los pines de entrada y salida se puede acceder y controlar desde el

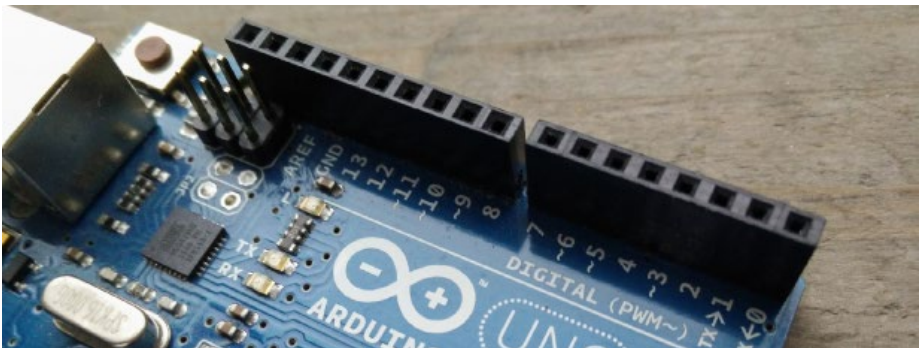


Pins de entrada y salida en la placa UNO cercanos al largo del chip microcontrolador

código escrito y cargarlo sobre la placa.

Los pines de cabecera también permiten la conexión a una constante de 5V y una fuente de alimentación de 3.3V (5 y 3,3 voltios). Estos simplemente le permitirán utilizar a UNO como una batería para alimentar sus componentes. En las baterías regulares se encuentran dos polos: positivo y negativo. Los pin 5V y 3.3V en el Arduino son los polos positivos, y los pines GND(de tierra) son los negativos.

Al lado de los pines de cabecera se encuentra el puerto USB tipo B utilizado para comunicarse con un PC y poder cargar el código. El conector en la parte inferior izquierda es donde se puede conectar una batería de 9V o cualquier otra fuente de alimentación de DC de entre 6 y 20V (lo recomendado es 7 -12 V), esto puede ser cualquier adaptador de red con el tamaño correcto del enchufe y cantidad de voltaje . A efecto de los siguientes

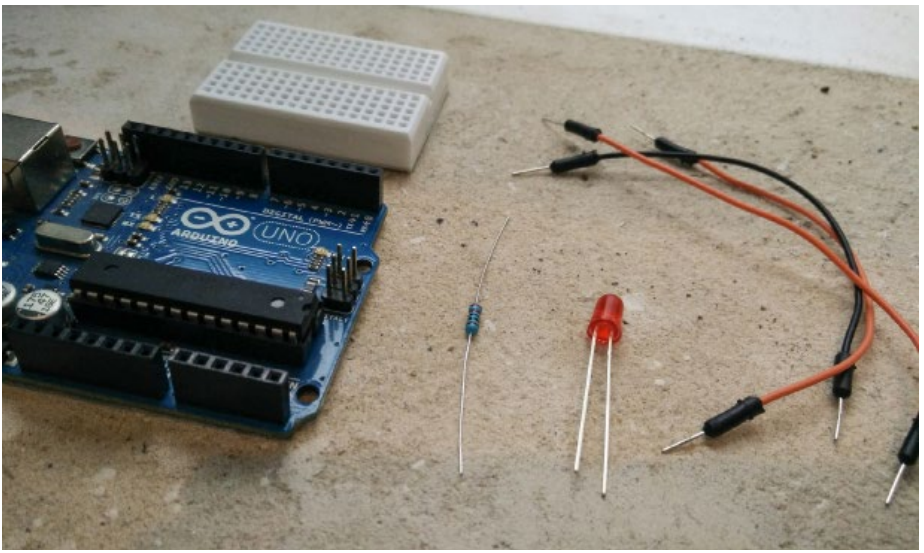


Pines de salida digitales

proyectos, el Arduino no será alimentado por una fuente externa, pero tendrá que ser conectado a través de USB a un ordenador, de donde utilizará la energía para el dispositivo.

Los pines de salida de la placa Arduino son también fuentes de energía, al igual que los pines son 5V. La diferencia es que los pines de salida se pueden activar y desactivar el chip microcontrolador y la aplicación que cargamos y corre sobre él.

Para demostrar el comportamiento de los pines de salida vamos a construir un circuito simple y controlarlo desde JavaScript. El "Hello World" de desarrollo de hardware parpadea un LED, así que eso es

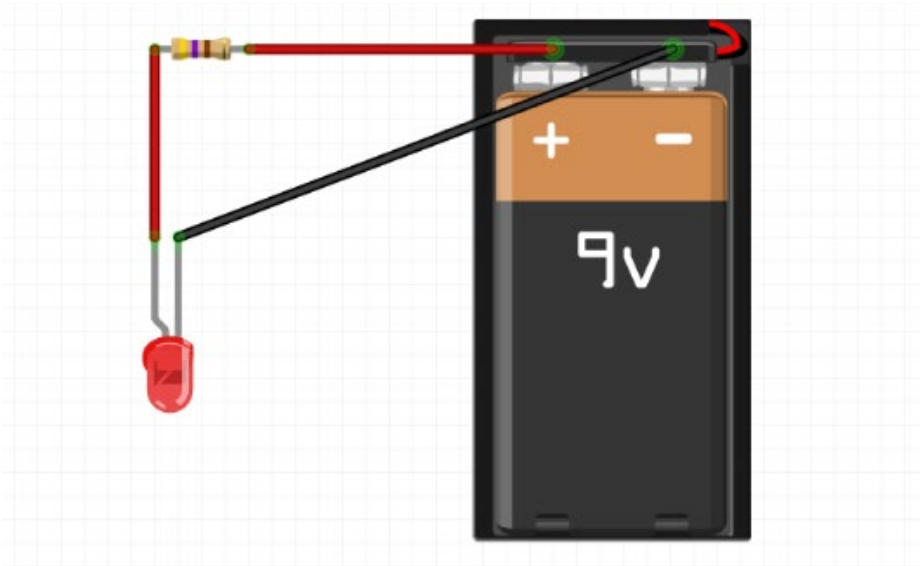


Componentes para el ejemplo de parpadeo de un LED

lo que vamos a hacer. Para este proyecto se necesita un Arduino UNO, un LED, una resistencia de 150 a 1k, una placa y algunos cables de puente.

Las resistencias pueden ser identificados a partir de las rayas de colores en su cuerpo. Hay muchas calculadoras de resistencia[1] y algunos trucos que sirven de ayuda.

El circuito más simple posible que podemos pensar sería un LED iluminado por una batería. Este circuito se conecta el polo positivo de la batería con la punta positiva del LED y el polo negativo con la punta negativa.

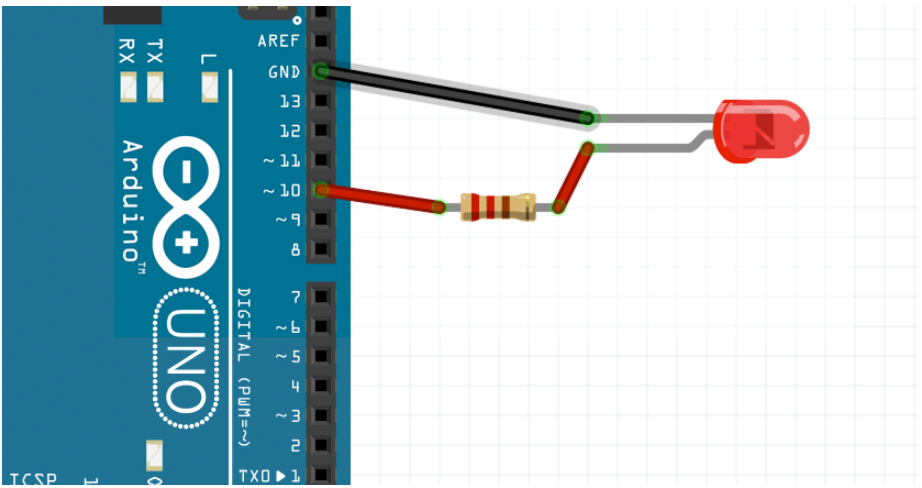


Batería y un LED (Fritzing.org)

[1] http://www.dannyg.com/examples/res2/resistor.htm

Una vez que se conectan, la electricidad y los electrones comienzan a fluir a través del led que inevitablemente provoca que se encienda.

La reconstrucción de este circuito simple en el Arduino será esencialmente significan reemplazar el polo positivo de la batería con el pin de 5V o uno de los pines de salida del Arduino y el polo negativo con GND. En este ejemplo, escogí al azar de salida número de pin 10.



LED conectado al pin de salida de Arduino (Fritzing.org)

Este circuito ya está completo y al cambiar al pin de salida 10 desde nuestro código el LED se iluminará.

Probablemente habrá notado la adición de una pequeña resistencia al circuito y se pregunta por qué

necesitamos esto. Bueno, hay un problema: el LED no limita la corriente eléctrica en nuestro circuito así que sin ella trataría de utilizar demasiada. Sin resistencia en nuestro circuito el LED se sobrecargaría y sufriría daños en muy poco tiempo. Probablemente sólo funcionaría durante un par de minutos antes de que se queme de forma irreversible.

Para evitar que esto suceda, podemos limitar el flujo de corriente eléctrica a través del LED mediante la adición de una resistencia. Usando la ley de Ohm [1] (resistencia es igual a la tensión dividida por la corriente o R = V / I) se puede calcular la resistencia exacta necesaria en una determinada instalación.

EEn nuestro circuito Arduino proporciona 5 voltios pero no todos que pasarán por resistencia: el LED también tiene una pérdida de tensión 2 voltios, que es la tensión suministrada ideal para que funcione. También sabemos que la corriente ideal para nuestro LED es 20 mA ó 0,02 en amperios. Ahora tenemos que elegir una resistencia que se reduzca la tensión en 20mA. El valor de la derecha viene dada por la ley de Ohm:

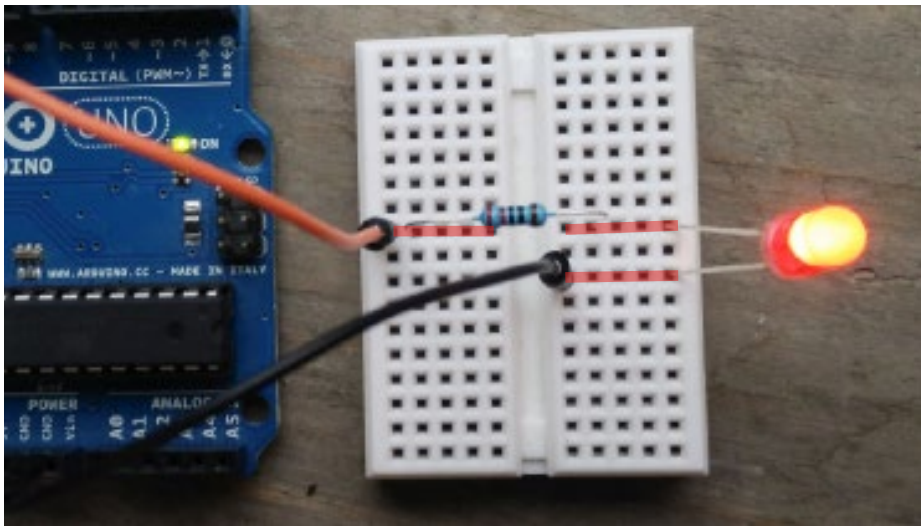
```
R = V / I
R = (5V - 2V) / 0.02A
R = 150Ω
```

[1] https://youtu.be/-mHLvtGjum4

Esto significa que necesitamos una resistencia alrededor de 150Ω. No queremos una resistencia que sea menos de 150Ω sino una más potente, por ejemplo 220Ω, 500Ω o 1000Ω, resistiría el flujo aún más, lo que haría que nuestro LED más tenue, pero a cambio extendería su vida.

Creando prototipo con placas de circuitos

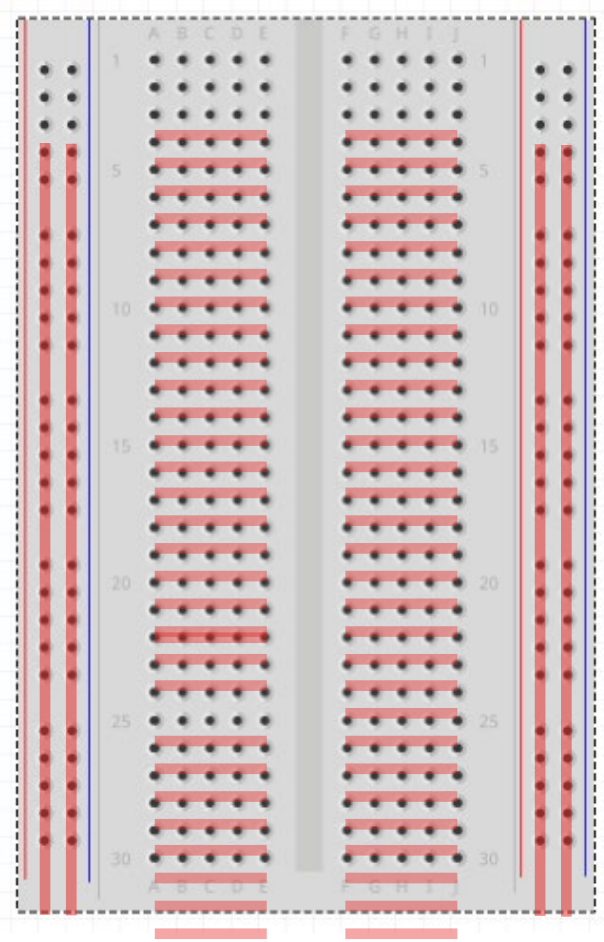
Las placas de circuitos(Breadboards) hacen que sea fácil y rápido construir prototipos de circuitos. Están diseñadas para conectar un macho a otra conexión de cable macho o a la punta de componentes simples como los LED, resistencias, diodos o capacitores. Las placas esencialmente ayudan conectando estos componentes entre ellos. Cuando conectamos la punta de un LED a cualquier lugar de lo placa, los componentes añadidos al



Circuito LED conectado en el interior de la placa de circuito

mismo pin estarán conectados al LED.

Aquí una placa de circuito más grande con todos sus conexiones pin resaltadas:



Pines conectados a una placa de circuito

EMPEZANDO CON NODE.JS

El LED está ahora listo para ser encendido y apagado mediante JavaScript. Para ello tendremos que ejecutar JavaScript en el servidor, con el fin de tener acceso al puerto USB que vamos a conectar con el Arduino. El primer paso es instalar Node.js en el ordenador.

La Instalación de Node.js es un proceso rápido y fácil, y se puede hacer con alguno de los instaladores oficiales ya desarrollado. Puede descargar estos desde https://nodejs.org/download/.

Una vez instalado, el código node estará disponible desde la Terminal en OS X o desde el símbolo del sistema en Windows. Para una demostración Hello World crear un nuevo archivo llamado helloworld.js y agrega esta sola línea de código JavaScript a la misma:

```
console.log(”Hola Mundo”);
```

Escribiendo `node helloworld.js` desde el terminal(desde la misma carpeta que contiene el archivo helloworld.js), ejecutará el archivo JavaScript y mostrará "Hola Mundo" en la línea de comandos.

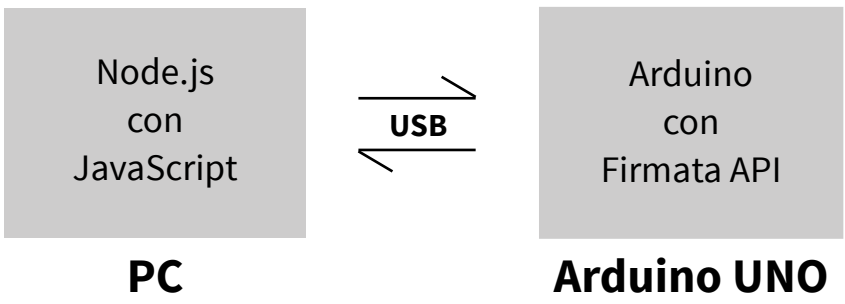
Node.js "Hola Mundo"

Esto puede no parecer demasiado avanzado, pero con la configuración de Node.js estamos listos para empezar a construir nuestros proyectos de hardware utilizando JavaScript. Desde este entorno, ahora tenemos acceso al puerto USB, sensores y muchos otros componentes en nuestro ordenador.

Conectando Node.js con Arduino

Para esta aplicación vamos a utilizar la librería Johnny-Five JavaScript[1] diseñada para Node.js. Esta librería fue creado por el equipo de Bocoup,[2] para hacer prototipos de hardware más fácil para los desarrolladores web.

Ejecutando código JavaScript directamente en placas de desarrollo sólo funciona con un grupo de modelos y por desgracia, el Arduino no es uno de ellos. Arduinos requieren escribir código en el lenguaje de Arduino, y esto actualmente no se puede cambiar. Johnny-Five, y muchas otras librerías de Node, ayudan a



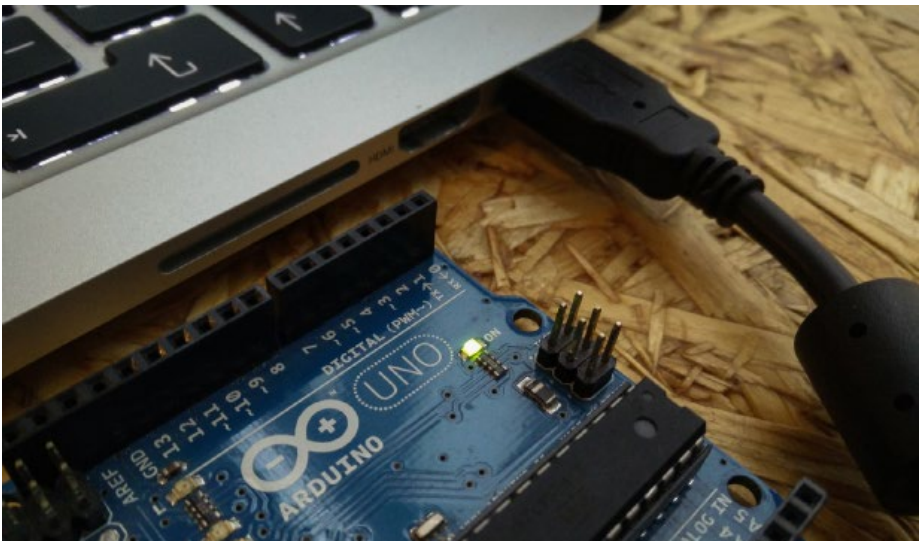
Función de Firmata

[1] https://github.com/rwaldron/johnny-five

[2] https://bocoup.com/

evitar esta limitación mediante la instalación de Firmata en Arduino, que esencialmente expone una API para comunicarse y controlar el Arduino desde dispositivos externos.

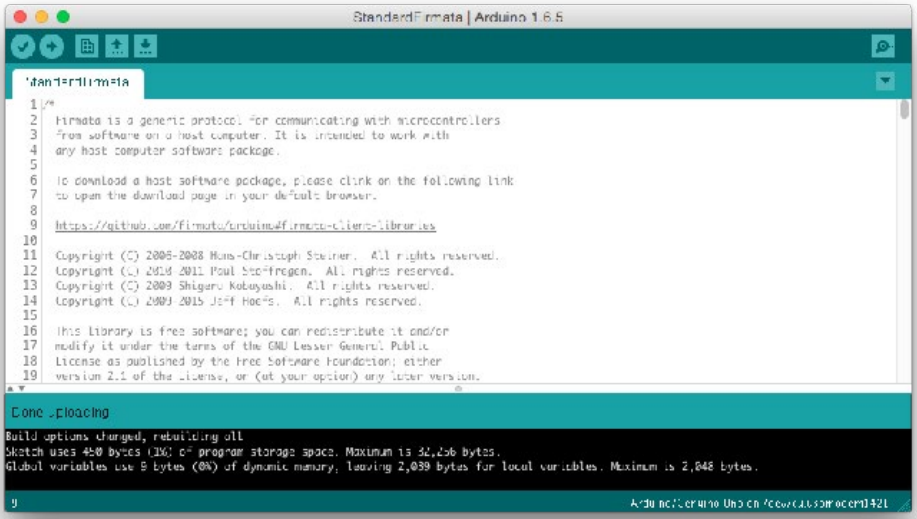
Para hacer parpadear el LED, primero tenemos que cargar la librería Firmata sobre la Arduino UNO[1]. Para cargar cualquier código en la placa tenemos que descargar e instalar el IDE oficial de Arduino de la web de Arduino. El IDE de Arduino es sólo el nombre de la aplicación que se utiliza para escribir y cargar el código de Arduino en las placas.



El Arduino UNO conectado al puerto USB

[1] https://www.arduino.cc/en/Main/Software

Una vez instalado, conecta tu Arduino en el puerto USB y el LED de alimentación verde, etiquetado con ON, debe encenderse. Abre la aplicación IDE que acaba de instalar y asegúrese de que bajo la ruta Herramientas/ Placa Arduino(Tools/Board Arduino) está seleccionado UNO. También tendrá que seleccionar el puerto USB que el Arduino se conecta en Herramientas / puerto serie(Tools/Serial Port). El puerto se mostrará en la lista cuando la placa está conectada. Si tiene problemas para conseguir con que la conexión funcione consulte la [guía de instalación oficial](#)[1]



IDE Arduino con el boceto Firmata subido

[1] https://www.arduino.cc/en/Guide/HomePage

o la página de solucionar problemas.[2]

A continuación es lo que usted verá cuando usted abre el IDE. En esta etapa no escribiremos nada en lenguaje Arduino. Sólo abrimos el IDE para cargar la librería estándar Firmata. Seleccione Archivo / Ejemplos / Firmata / StandardFirmata(File/Examples/Firmata/StandardFirmata) para abrir el boceto y luego presione cargar(upload la flecha verde que apunta a la derecha), esto cargará la librería Firmata a tu UNO.

Si todo se ha realizado correctamente, verá el mensaje "Done Uploading." En la barra de estado verde. Si recibe un mensaje de error, asegúrese de que el Arduino está conectado, tiene corriente, y que se ha seleccionado la placa y el puerto correcto en el menú de herramientas.

Si tuviera algún problema para instalar la librería Firmata, diríjase a la guía de instrucciones.[1]

[1] http://www.instructables.com/id/Arduino-Installing-Standard-Firmata

[2] https://www.arduino.cc/en/Guide/Troubleshooting

PARPADEO DE UN LED

El cableado ya está terminado y el Arduino tiene la librería Firmata cargada. El UNO está listo para tomar órdenes desde Node.js.

Siéntase libre de descargar el código fuente de todos los ejemplos de www.webondevices.com/download-source/ si lo prefiere para continuar de esa manera.

Como vamos a utilizar la librería Johnny-Five para parpadear nuestro LED, vamos a instalarlo. Asegúrese de crear una nueva carpeta y navegar a la misma desde la línea de comandos antes de ejecutar el comando de instalación a continuación:

```
npm install johnny-five --save
```

Vamos también a crear un nuevo archivo para nuestra aplicación Node.js y el nombre de blink.js. Lo primero que tenemos que hacer es cargar todas

las liberías en las variables requeridas por nuestra aplicación. En nuestro caso es sólo Johnny-Five:

```
var five = require("johnny-five");
```

Lo siguiente que haremos es inicializar una instancia nueva para la placa que nos proporcionará todos los métodos para interactuar con Arduino:

```
var arduino = new five.Board();
```

Cuando se trabaja con jQuery usamos el `$(document).on("ready", callback);` como detector del eventoque espera hasta que el documento se ha terminado de cargar antes de hacer cualquier otra cosa. El Arduino, y la conexión USB también necesitan algo de tiempo para ponerse en marcha, por lo que necesitamos implementar un detector de eventos similar antes de enviar nuestros comandos.

```
arduino.on("ready", function(){
    // La placa está lista. Ahora
    // podemos hacer parpadear el LED
});
```

Para hacer parpadear el LED, en primer lugar, vamos a necesitar crear una instancia de LED dentro de la devolución de llamada(callback) del detector de eventos:

```
// Crear un LED en el número de pin 10
var led = new five.Led(10);
```

Entonces, debajo de ella, sólo podemos llamar blink():

```
// Parpadear los LED cada segundo
led.blink(500);
```

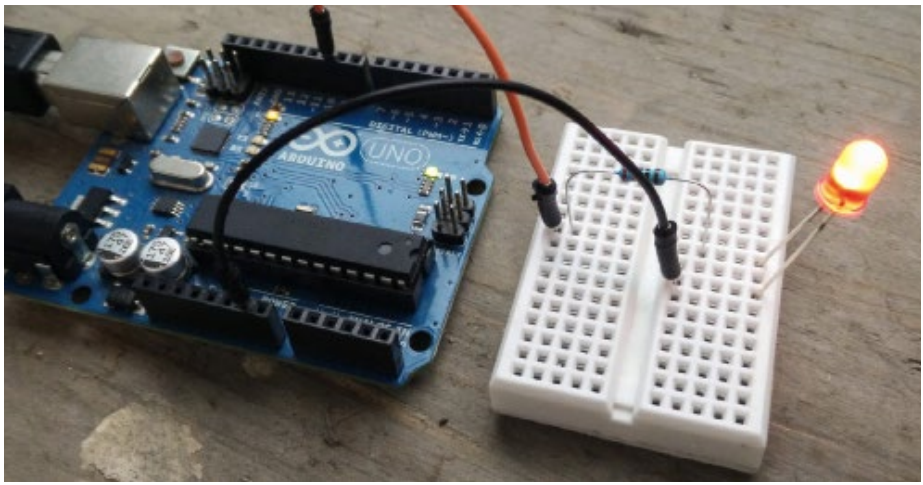
Aquí está el código completo:

```
var five = require("johnny-five");
var arduino = new five.Board();

arduino.on("ready", function(){
    var led = new five.Led(10);
    led.blink(500);
});
```

Ahora que el LED está configurado podemos intentar otros métodos también: led.pulse (); apagará y encenderá el led gradualmente(efecto fade) en lugar de simplemente encenderlo y apagarlo sin transición alguna. Convenientemente, métodos IN y OUT están disponibles:

```
// encender gradualmente el led
led.fadeIn();

// Esperar 3 segundos y luego
// apagar gradualmente
```

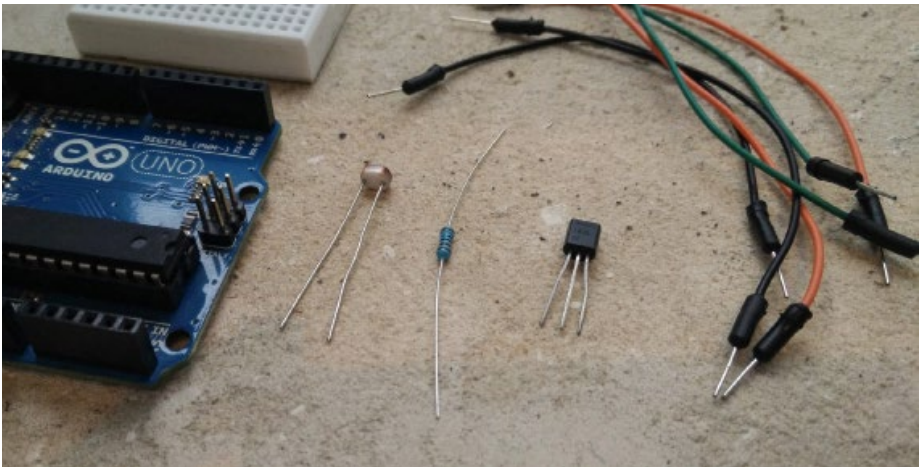
```
this.wait(3000, function(){
    led.fadeOut();
}
```



Parpadeando un LED desde el pin de salida número 10

SENSORIZANDO EL MUNDO

En este capítulo, vamos a ampliar nuestro conocimiento mediante la exploración de los pines de entrada. Ya sabemos que estos pines pueden medir el cambio en la tensión por lo que vamos a utilizar sensores analógicos simples que afectan a la tensión/voltaje a través de ellos en función de ciertos cambios en el entorno.



Componentes para los proyectos con sensores

Para este proyecto, se necesita un Arduino UNO, un sensor de temperatura LM35 o TMP36, un LDR, una placa de circuitos, un dos resistencias de 1k y algunos cables de puente.

Midiendo la temperatura

Para medir la temperatura utilizaremos el bastante común sensor LM35. Este sensor de bajo coste opera entre -55 y 150 ° C, con una precisión de +/- 0,5 ° C (aunque es un poco difícil de medir temperaturas bajo cero con el LM35). La forma en que este sensor analógico funciona es muy simple y de hecho, la mayoría de los sensores analógicos funciona de una manera similar. En primer lugar se alimentan de una fuente de energía constante en dos de sus pines (+ y -) y, a continuación, en un tercer pin, dan salida a un valor de tensión inferior que es directamente proporcional a la lectura del sensor.

Nuestro sensor de temperatura se alimentara del pin de 5v de Arduino UNO y tendrá una salida entre 0 y 2 voltios, que cambiará con la temperatura. El factor de escala de LM35 es 0.01V / ° C, lo que significa que por cada grado centígrado en la temperatura del aire el voltaje cambiará en 0.01 voltios en el pin de salida.

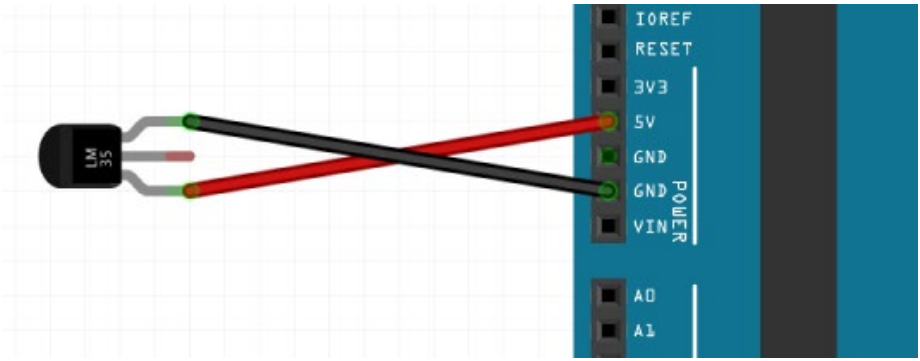
Convenientemente, los pines de entrada analógica de la Arduino están diseñados para medir y convertir la tensión en datos que podemos utilizar. En números reales se asignará el rango de 0 a 5 voltios que puede ser interpretado por los pines analógicos de la Arduino UNO a una escala de 0 - 1024 en nuestro código del programa.

Digamos que su sensor devuelve sólo 1.5 voltios de los suministrados 5 voltios al pin de entrada analógica del cual está conectado. La lectura en su aplicación será 307 en la escala de 0 - 1024 porqué (1.5 / 5) * 1024 es 307,2.

La librería Johnny-Five mapea de 0 a 1024 en escala de medida de -55 a 150 grados centígrados y luego calcula la temperatura resultante de esta información.

Volvamos a nuestro sensor de temperatura y lo cableamos. Antes de comenzar el cableado, asegúrese siempre de haber desenchufado el Arduino desde el USB, o cualquier otra fuente de energía. De lo contrario se podría meter accidentalmente un cable en la clavija que equivocado que podría dañar la placa, o el sensor.

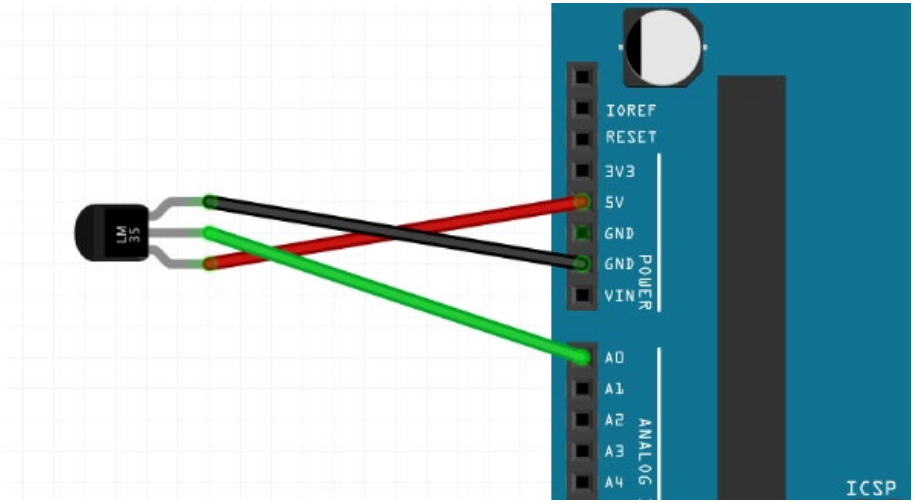
De las tres puntas del sensor la de la izquierda es el positivo y el de la derecha es el polo negativo (el lado plano del sensor es la parte delantera). Vamos a alimentar con los pines 5V (positivo) y GND (negativo) del Arduino UNO como se ha visto en la página anterior.



Sensor de temperatura conectado (Fritzing.org)

El sensor está alimentado, y el único pin que nos queda es el de salida. Esto tendrá que ser conectado a una de las cinco pines analógicos del UNO. Estos son los que se alinearon a lo largo del borde inferior derecho de la placa, marcado A0 a A5.

Vamos a conectarlo al pin A0:



Sensor de temperatura conectado al pin de salida del Arduino UNO (Fritzing.org)

Ahora que el cableado esté terminado, vamos a escribir el código que leerá el sensor. En primer lugar, creamos un nuevo proyecto y el nombre del archivo principal temp.js, comenzando de la misma manera que el boceto de parpadeo:

```
var five = require("johnny-five");
var arduino = new five.Board();

arduino.on("ready", function(){
    // El Arduino está listo
});
```

A continuación, tenemos que crear una nueva instancia del sensor de temperatura con algunos ajustes, que son el nombre del sensor, y el número pin está conectado.

```
var tempSensor = new five.Temperature({
    controller: "LM35",
    pin: "A0"
});
```

Si dispone un sensor TMP36 también se puede utilizar de la misma manera. Basta con cambiar los ajustes del controlador a TMP36 y el cableado es el mismo.

Después de inicializar el sensor de temperatura ya podemos empezar a recoger los datos a través del listener de eventos a medida que van llegando. La librería Johnny-Five aprovecha al máximo las capacidades asíncronas de Node.js lo que significa que las lecturas del sensor aparecen inmediatamente en la función de devolución de evento de datos del oyente.

Vamos a escribir este detector de eventos:

```
tempSensor.on("data", function(er, data){
    console.log(data.celsius + "°C");
    console.log(data.fahrenheit + "°F");
});
```

Una vez más, tenemos que esperar a la placa para ser inicializado por lo tanto estos bloques tendrán que ir a la función `arduino.on("ready", callback);`.

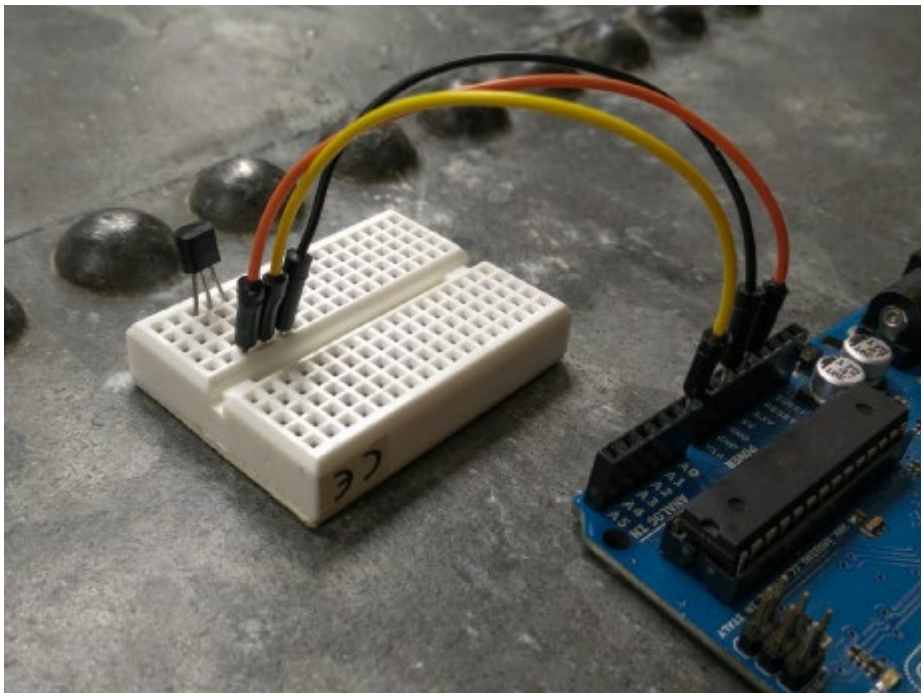
En la versión final del script que se encuentra en la página siguiente, tanto los lectores de grados centígrados y fahrenheit redondean a un decimal para que los números sean fáciles de leer.

```
var five = require("johnny-five");

var arduino = new five.Board();

var celsius = 0;
var fahrenheit = 0;

arduino.on("ready", function(){
```

```
var tempSensor = new five.Temperature({
    controller: "LM35",
    pin: "A0"
});

tempSensor.on("data", function(er, data){
    celsius =
        data.celsius.toFixed(1);
    fahrenheit =
        data.fahrenheit.toFixed(1);
    console.log(celsius + "°C");
    console.log(fahrenheit + "°F");
});

});
```

Ejecute el script introduciendo `node temp.js` en la línea de comandos. Compruebe las lecturas, y si hay anomalías, vuelva a comprobar el cableado, asegúrese de que el Arduino está conectado al puerto USB, y la luz verde está encendida. Si todo está bien, podemos pasar al sensor de luz.

Aquí está el circuito de temperatura LM35 añadido a una pequeña placa:



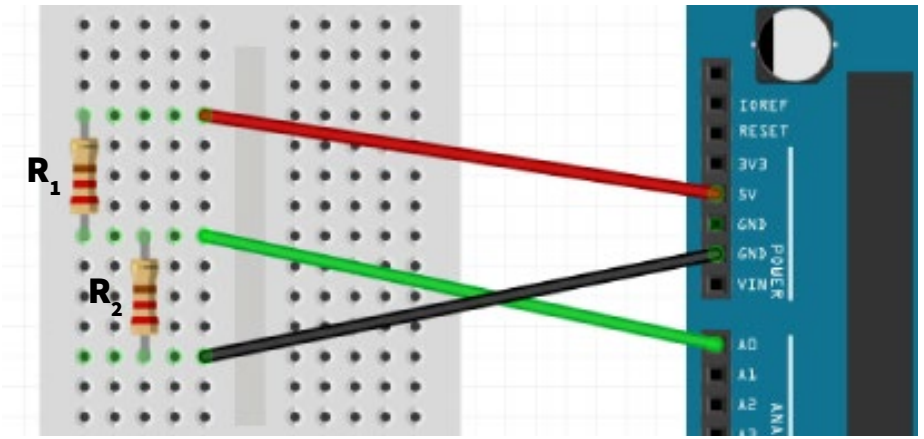
Circuito de sensor de temperatura en una placa de circuitos

Midiendo la luz

Nuestro sensor de luz es uno de los sensores más simples disponibles, conocidas como LDR (por sus siglas en inglés), que significa Resistencia de Luz Dependiente. Un LDR actúa como una resistencia regular para el flujo de corriente. La principal diferencia es que el LDR cambia su resistencia en función de las condiciones de luz.

Lamentablemente la resistencia no es fácil de medir para los campos de entrada analógicas de la Arduino. Para convertir este cambio de resistencia a cambio de voltaje, ya que los campos de entrada son fáciles de medir, tendremos que construir un circuito divisor de tensión simple. Un circuito divisor de tensión divide una tensión más grande en otras más pequeños con la relación de las dos resistencias.

La entrada de 5 voltios de este circuito va desde la placa Arduino a través del cable rojo. La tensión de salida a través del cable verde es directamente proporcional a la tensión de entrada y la relación de las resistencias (R1, R2).



Divisor de tensión (Fritzing.org)

$$V_{out} = V_{in} * (R_2 / (R_1 + R_2))$$

En este circuito se utilizan dos resistencias de 1000Ω así que aquí está cómo sería la ecuación con estos valores:

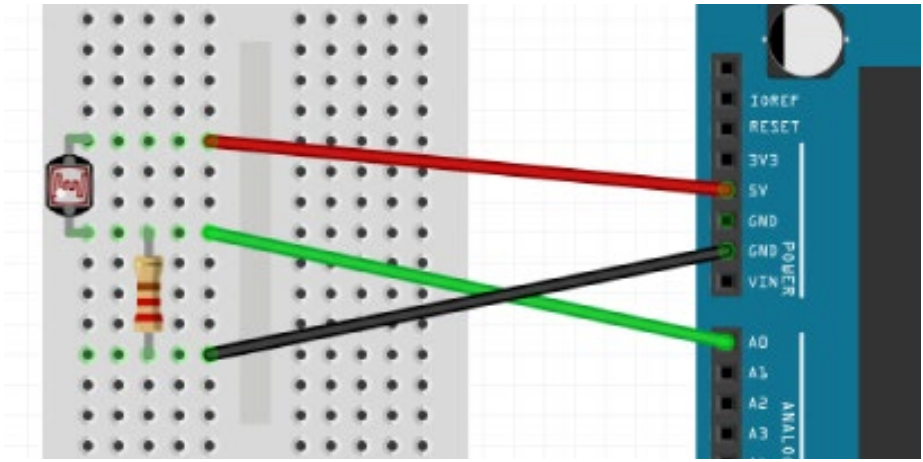
$$V_{out} = 5V * (1000 / (1000 + 1000))$$
$$V_{out} = 5V * 0.5$$
$$V_{out} = 2.5V$$

Esto significa que si dos de las mismas resistencias se utilizan en un circuito divisor de tensión la tensión de salida será la mitad del voltaje de entrada. 2,5 voltios en nuestro caso. Usando la ecuación también vemos que cambiando sólo una de las resistencias

cambiará la tensión de salida hacia arriba o abajo.

En el circuito LDR cambiamos una de las resistencias habituales de la resistencia dependiente de la luz. La resistencia del LDR cambiará con las condiciones de luz en el circuito divisor de tensión que a cambio cambiará constantemente la tensión de salida para nuestro pin de entrada.

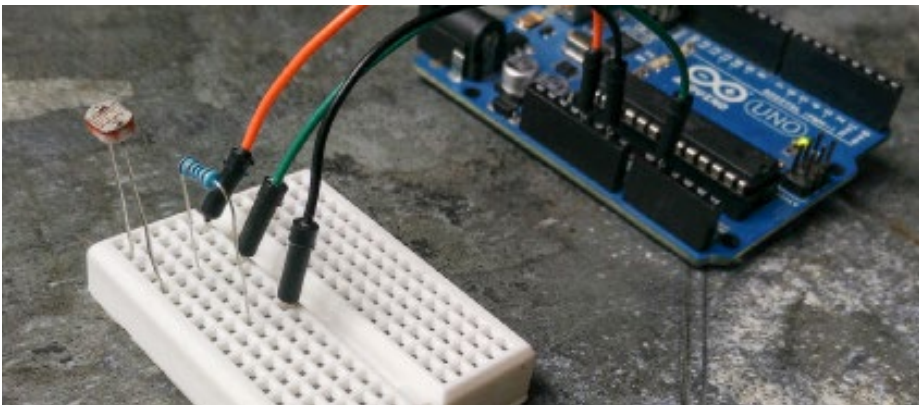
Cuando la foto-resistencia se expone a la luz, su resistencia disminuye por lo que la lectura de la tensión será mayor. Por el contrario, con menos luz la lectura de voltaje será menor. El valor cambiante de la tensión es entonces el calculada por los pines de entrada analógicas del factor de Arduino.



Circuito de sensor de luz (Fritzing.org)

La resistencia dependiente de la luz cambia su resistencia entre 150Ω con luz brillante a 9000Ω en la oscuridad. Manteniendo fija la resistencia original de 1000Ω y utilizando la ecuación de división de tensión podemos decir que nuestra tensión de salida cambiará entre 4.35V y 0.5V lo cual es una gran gama para los pines de entrada de Arduino.

Así es como quedaría el circuito en la placa:



Circuito de sensor de luz (Fritzing.org)

El circuito está terminado así que vamos a agregar el sensor de luz a nuestra aplicación JavaScript en la función de callback del evento inicializador(ready) de arduino. Inicializamos un nuevo LightSensor y continuación agregamos un evento en datos oyente(listener):

```
var lightSensor = new five.Sensor({
    pin: "A1",
    freq: 250
});

lightSensor.on("data", function(){
    console.log(this.value);
});
```

Este código es muy parecido a la forma en que el sensor de Temperatura es usado por la librería Johnny-Five. En primer lugar, un nuevo sensor necesita ser inicializado con algunas opciones de configuración, a continuación, la instancia del sensor en el listener de eventos de datos se utiliza que esperar la llegado de datos.

Aquí está la versión final del código de sensor de luz:

```
var five = require("johnny-five");

var arduino = new five.Board();

var light = 0;
```

```
arduino.on("ready", function(){

var lightSensor = new five.Sensor({
    pin: "A1",
    freq: 250
});

lightSensor.on("data", function(){
    light = this.value;
    console.log(light);
});

});
```

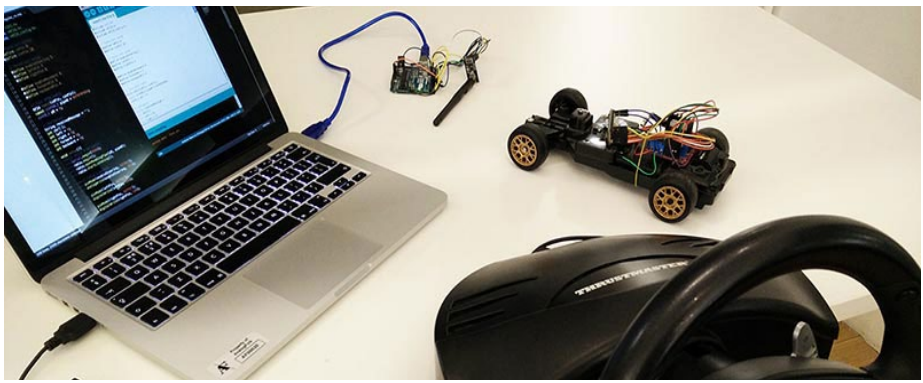
Guardando esto en un nuevo archivo light.js y tecleando `node light.js` desde la línea de comandos dará lugar a lecturas de los sensores de luz que aparecen cada cuarto de segundo. la propiedad Freq (frecuencia) es donde podemos cambiar este comportamiento que espera un valor de milisegundos para definir la demora entre las mediciones.

Descargar el código fuente de todo el proyecto de www.webondevices.com/download-source

SEGUIR AVANZANDO

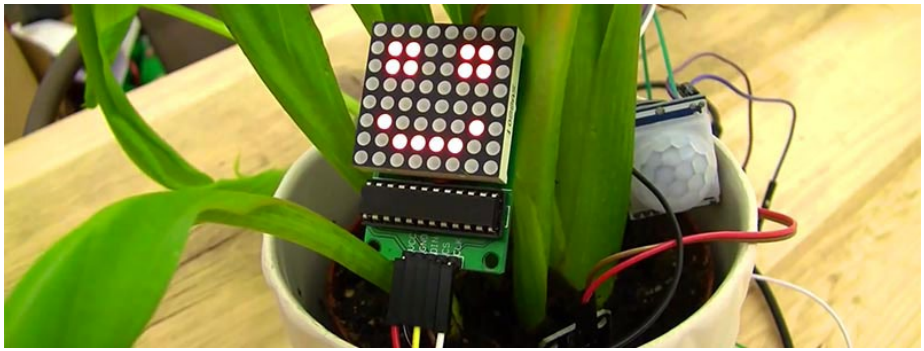
En este manual hemos aprendido los conceptos básicos del uso de Arduino UNO usando Node.js a través del puerto USB. En Web on Devices, www.webondevices.com, se encuentran varios proyectos construidos usando métodos similares.

Echa un vistazo al coche teledirigifo [1] que fue reconstruido desde cero utilizando Arduino. Se conecta a un ordenador de forma inalámbrica y con la API Gamepad permite al usuario conducir el coche con un volante USB.

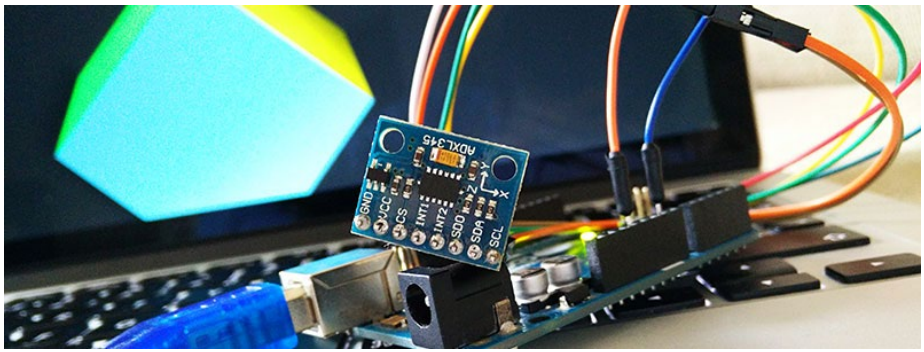


[1] http://www.webondevices.com/arduino-nodejs-rc-car-driven-with-html5-gamepad-api/

Conoce a George, la planta que habla. A través de sus sensores puede detectar la temperatura[1], la luz, el movimiento y la humedad del sustrato de su maceta. Se queja cuando no está contento con cualquiera de las lecturas de los sensores, y también puede responder a preguntas básicas. Es capaz de hablar y escuchar utilizando la API WebSpeech.



También podrás encontrar el proyecto giroscopio Arduino[2] que permite rotar objetos CSS en 3D en pantalla utilizando un controlador físico.



[1] http://www.webondevices.com/the-arduino-plant-with-javascript-voice-recognition/

[2] http://www.webondevices.com/rotate-a-css-3d-cube-with-an-arduino/

Usar Node.js para comunicarse con Arduino es sólo una de las muchas maneras de utilizar JavaScript para la construcción de proyectos electrónicos. En estos ejemplos, el código JavaScript se ejecuta en el procesador del ordenador, y luego ejecutamos comandos hacia Arduino. Otras placas como el Raspberry Pi, Arduino Yun, Tessel y el Espruino realidad pueden ejecutar JavaScript por ellos mismos. Las placas Particle, compatibles con Arduino, presentan una API RESTful, y existe una librería de Node.js para trabajar con ellos también. Las placas Particle pueden conectarse a Internet de forma inalámbrica, por lo que Node no depende de una conexión USB.

El proyecto Web on Devices se dedica a seguir experimentando los límites de lo que es posible con placas de desarrollo y dispositivos inteligentes que aprovechan las tecnologías web. En los próximos proyectos vamos a explorar todas las tablas y las técnicas mencionadas.

Siga Web on Devices en Facebook, Twitter o Instagram para no perderte ninguno de nuestros interesantes futuros proyectos:

www.facebook.com/webondevices
www.twitter.com/web_on_devices
www.instagram.com/web_on_devices

Web on Devices

Integrando electrónica con Javascript y otras tecnologías Web

www.webondevices.com