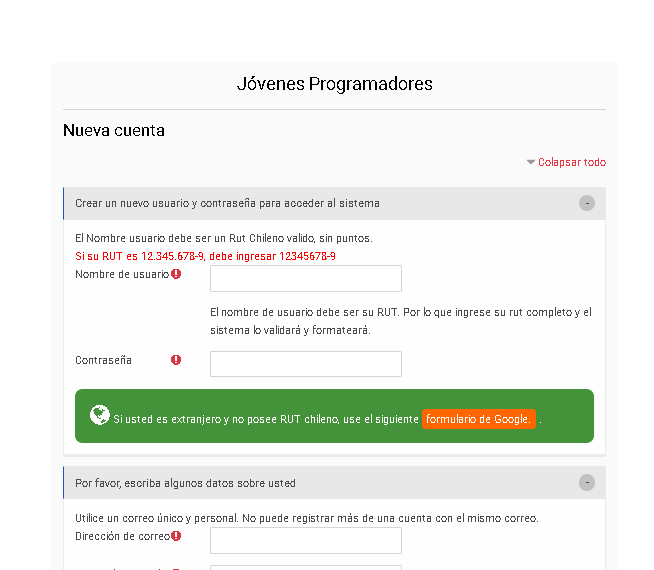
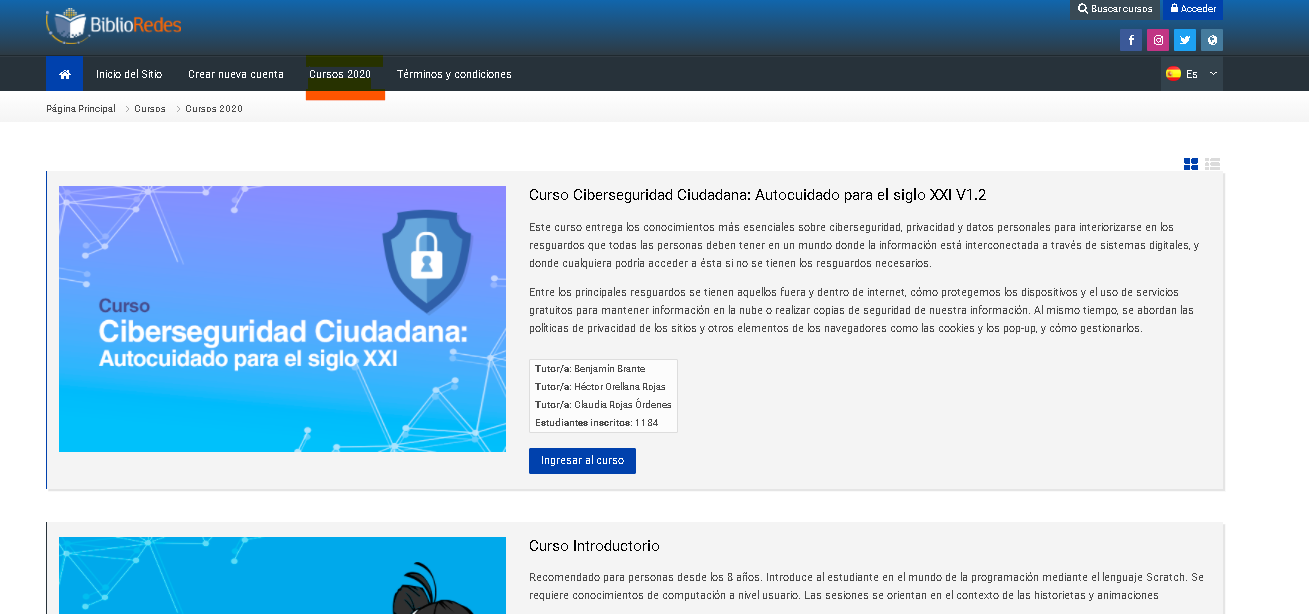
**RECURSOS DE APOYO PARA CREACIÓN DE INTERNET DE LAS COSAS**

Si estás leyendo esto, es porque tu grupo de trabajo decidió realizar una solución tecnológica relacionada con **Internet de las Cosas**. La siguiente guía entrega material suficiente para el desarrollo de soluciones tecnológicas basadas en Arduino. Pueden leer en forma individual o en equipo.

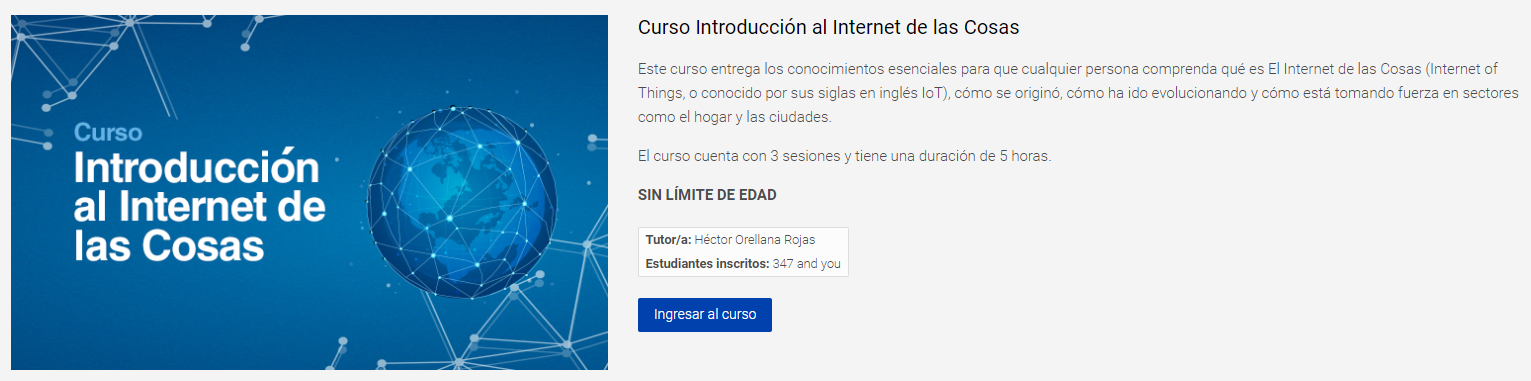
1. En primer lugar, deberás crearte una cuenta en Jóvenes Programadores, haciendo click en el siguiente [enlace](https://jprogramadores.biblioredes.gob.cl/login/signup.php?).



1. Luego, debes acceder con tus contraseñas creadas y dirigirte a la sección cursos 2020 que se muestran a continuación.



1. Deberás buscar el curso “Introducción a Internet de las Cosas” y matricularte.



1. Una vez que completes el curso deberás tomar estos dos cursos:
   1. Curso Javascript 1
   2. Curso Javascript 2

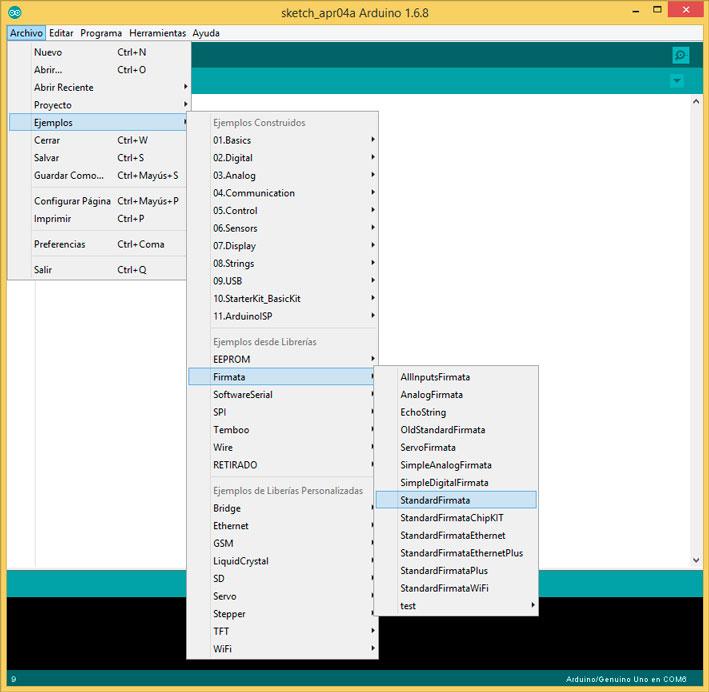
Terminado estos cursos, estarás capacitado para para crear soluciones tecnológicas usando Arduino y Javascript. Aquí es donde ustedes se preguntan ¿Por dónde empiezo?. No se preocupen, a continuación se presentará una guía extraída de la web [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/) que permite controlar Arduino vía Web con Firmata y Javascript.

**¿Qué es Firmata?**

Es un protocolo que permite comunicar microcontroladores desde cualquier software o cualquier ordenador conectado. El objetivo principal es lograr que el microcontrolador sea una extensión del entorno de desarrollo. Esto permite programar en diversos lenguajes de programación, enfocándose en herramientas y kit de desarrollo para controlar los dispositivos. Actualmente existe una [gran cantidad de librerías y frameworks](https://github.com/firmata/protocol#firmata-client-libraries) compatibles para la programación de Arduino y sus microcontroladores, sin embargo en esta guía nos centraremos en **johnny-five**, permitiendo el control de nuestro Arduino vía web con Firmata y Javascript.

**Configurar Arduino**

Lo primero que debes realizar es cargar el estándar Firmata en la placa Arduino. Para ello, conectamos nuestra placa Arduino con el cable USB al computador, abrimos el entorno de desarrollo oficial y vamos a: **Archivo > Ejemplos > Firmata > StandarFirmata**.



Fuente: [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/)

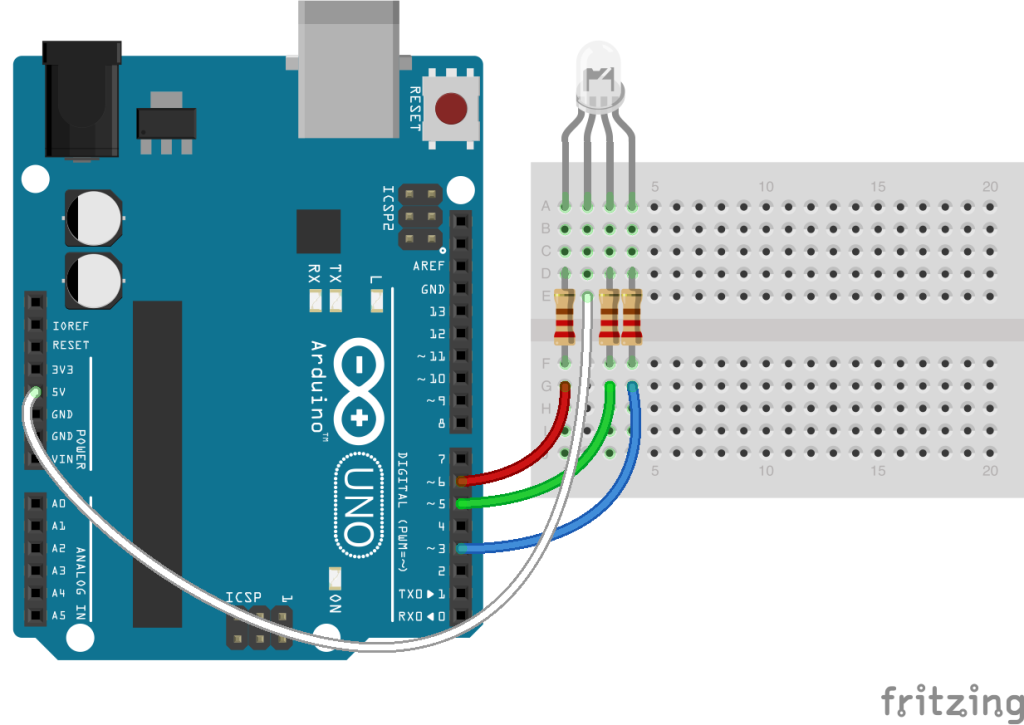
Esto abre un sketch con el código del protocolo Firmata. Solo nos queda subir dicho sketch a nuestra placa. Con esto ya tendríamos preparado el entorno de desarrollo y la placa para recibir los comandos del protocolo. En los siguientes pasos vamos a ver el esquema eléctrico y la aplicación que nos permita controlar Arduino vía web con Firmata.

**Conectando los componentes necesarios LED RGB y resistencias**

El proyecto va a consistir en controlar un LED RGB a través de una aplicación web. Este LED puede emitir luz de diferentes colores, mezclando los tres componentes básicos (Rojo, Verde y Azul) podremos obtener cualquier tonalidad. La aplicación va a ser muy sencilla, nos permitirá encender, apagar, parpadear, iluminar con color rojo, verde o azul.

Para controlar el LED RGB necesitamos utilizar 3 pines digitales PWM y la alimentación de 5V. Para no superar el voltaje de operación del LED RGB, debemos utilizar 3 resistencias de 220Ω en serie con las 3 patillas que van a los pines digitales.

El esquema de conexión de los componentes con la placa de Arduino sería el siguiente.



Fuente: [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/)

Puedes probarlo para comprobar su funcionamiento. Lo único que tienes que hacer es poner la salida digital correspondiente a un valor entre 0 y 255. Este es el rango de valores que soporta este tipo de salidas. Es importante comprobar que todo está bien conectado antes de controlar Arduino vía web con Firmata.

**Framework Johnny-five**

Como ya te he dicho, vamos a utilizar el framework [Johnny-five](http://johnny-five.io/). Esto nos facilitará la comunicación entre la aplicación y Arduino. El lenguaje que hablen entre ellos será Firmata, el protocolo estándar. Johnny-five es un framework de JavaScript y por lo tanto, debemos tener una nociones básicas con este lenguaje.

Si quieres profundizar más con Johnny-five, puedes echar [un vistazo a los ejemplos](http://johnny-five.io/examples/) de su web oficial. Encontrarás más información y ejemplos detallados de diferentes funcionalidades que aporta este framework.

Montar el servidor

Para este cometido vamos a utilizar [NodeJS](https://nodejs.org/es/), uno de los framework más populares para plataformas de desarrollo Back-End. Trabaja con una arquitectura orientada a eventos y utiliza como lenguaje de desarrollo Javascript aprovechando el motor v8 creado por Google. Instalamos o actualizamos desde el siguiente [enlace](https://nodejs.org/es/download/) (Si utilizas Linux puedes obtenerlo con el siguiente comando apt-get install nodejs).

Una vez instalado seguimos con el resto de componentes. Para el resto de herramientas vamos a utilizar un archivo llamado **manifiesto** donde le indicaremos las dependencias de nuestra aplicación. Creamos nuestra carpeta donde vamos a montar nuestra aplicación y en la raíz de esta ponemos el siguiente archivo que tenemos que llamar **package.json**:

|  |
| --- |
| {  **"name"**: "nodebot-sample",  **"description"**: "Ejemplo IoT",  **"author"**: "Autor",  **"version"**: "0.1.0",  **"dependencies"**: {  **"express"**: "^4.13.3",  **"johnny-five"**: "^0.8.53",  **"socket.io"**: "^1.3.3"  }  } |

En este archivo de configuración le estamos diciendo que nos instale [express](https://expressjs.com/), [socket.io](https://socket.io/) y [johnny-five](http://johnny-five.io/), de momento solo céntrate en Johnny-five que es la librería que nos interesa ahora mismo. Más adelante veremos la utilidad del resto de paquetes.

Debemos guardarlo en la carpeta raíz de nuestro proyecto por ejemplo, si has creado una carpeta que se llama “Proyecto IoT”, crea en ella un archivo que se llame package.json. Lo editas y copias el contenido de arriba. Luego lo guardas y ya lo tenemos listo.

Ahora solo nos faltaría ejecutar dicho archivo. Esto lo haremos a través de la línea de comandos o CMD. Lo puedes ejecutar de diferentes formas. si utilizas Windows 7, Ctrl + R y escribir cmd. Si utilizas Windows 10, en Cortana escribes cmd y te aparece el primero. En Linux o Mac a través de un terminal.

Una vez lo tengas abierto debes ir a la carpeta donde has guardado el package.json y ejecutar el siguiente comando.

|  |
| --- |
| npm install |

Esto instalará todas las dependencias que hemos indicado en nuestro archivo package.json. Si echas un vistazo a la carpeta de tu aplicación se habrá creado la carpeta node\_modules y dentro de ella estarán todas las librerías que vamos a utilizar. Como ves es muy sencillo. Ahora vamos a probar la potencia del framework Johnny-five.

**El primer sketch**

Lo primero que haremos es una prueba muy sencilla. Vamos a encender y apagar el LED que viene integrado dentro de la placa de Arduino, el típico hola mundo o blink. Si utilizas un Arduino UNO es el pin 13.

Abre [Visual Studio Code](https://code.visualstudio.com/) y crea un archivo Javascript. Copia el siguiente código.

|  |
| --- |
| var five = require("johnny-five"),board, led;    board = new five.Board();    board.on("ready", function() {  led = new five.Led(13);  led.strobe(1000);  }); |

Nombra a este archivo **app.js** y guárdalo en la raíz de la carpeta de trabajo que creaste.

La primera línea permite incorporar el framework johnny-five , permitiendo usar sus clases u objetos ya definidos.

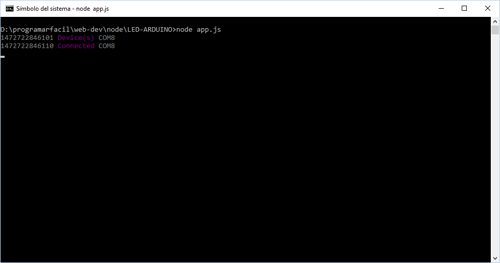
En la línea 3, se inicia la comunicación con la placa de Arduino y solo se encarga de buscar el puerto serie donde está conectado. Si tenemos dos Arduinos conectados al mismo ordenador, también podemos pasar como parámetro el puerto serie del Arduino que queremos utilizar.

En la línea 5, tenemos el método que se va a ejecutar una vez la conexión se realice con éxito. Es equivalente al método **void loop(){….} del código nativo de Arduino**. Aquí podemos hacer referencia a los diferentes componentes y pines del circuito.

Asegúrate de que el Arduino está conectado a través del cable USB, y ejecuta la siguiente línea de comandos en el terminal.

|  |
| --- |
| node app.js |

Cuando lo ejecutas, te informa de que se ha conectado a la placa. Dependerá del puerto que tenga asignado pero debe aparecer algo parecido a esto.



Fuente: [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/)

Inmediatamente el LED empezará a parpadear. Si quieres interrumpir la ejecución solo tienes pulsar a la vez **Ctrl + C**. Como ves resulta muy sencillo encender y apagar un LED con Jhonny-five. Te recomendamos a este punto de la guía investigar el framework Jhonny-five creando nuevas funcionalidades con tu Arduino.

Para controlar Arduino vía web con Firmata, vamos a necesitar alguna cosa más.

**Controlar Arduino vía web con Firmata y Javascript**

Socket.io es lo que se denomina un WebSocket y nos va a permitir una comunicación entre el navegador (la interfaz que vamos a construir) y el servidor (en este caso NodeJS). Al final lo que haremos será pasar mensajes y eventos para controlar a nuestro Arduino.

Para nuestra interfaz vamos a utilizar [Bootstrap](https://getbootstrap.com/) y HTML. Vamos a montar una sencilla web con varios botones, uno azul, otro verde y otro rojo para ir cambiando el color de nuestro LED RGB y otros tres botones para encender, apagar y hacer que parpadee nuestro LED. Veremos un simple ejemplo para utilizar Arduino vía web con Firmata y JavaScript.

En primer lugar, modifica el archivo app.js con el código necesario en la parte del servidor:

**Código Javascript**

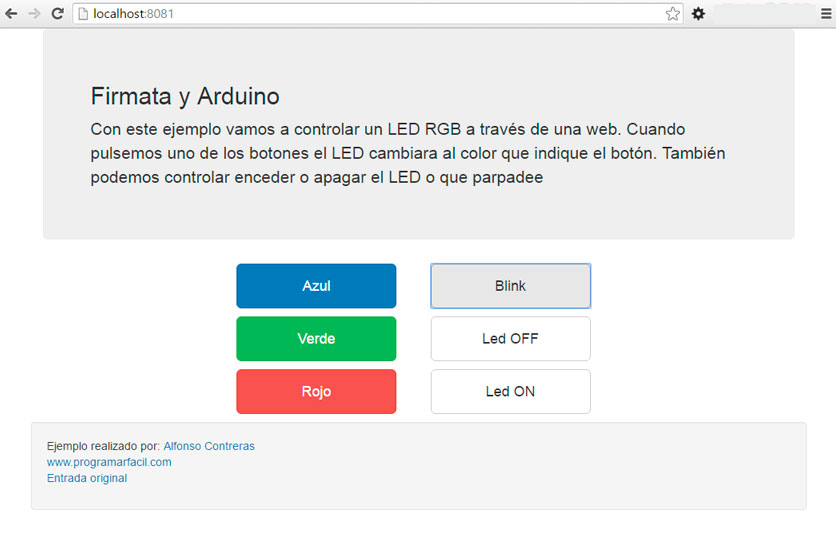
|  |
| --- |
| var express = require('express');  var app = express();  var io = require('socket.io')(app.listen(8081));  var five = require('johnny-five');    app.use(express.static(\_\_dirname + '/app'));    app.get('/', function (req,res) {  res.sendFile(\_\_dirname + '/index.html');  });      var board = new five.Board({  repl:false  });    board.on('ready', function () {  var speed, commands, motors;  var anode = new five.Led.RGB({  pins: {  red: 9,  green: 11,  blue: 10  },  isAnode: true  });    commands = null;    anode.on();  anode.color("#efe13d");    anode.blink(1000);    var blink = true;    io.on('connection', function (socket) {  socket.on('azul', function (){  anode.on();  anode.color("#3366CC");  });    socket.on('verde', function (){  anode.on();  anode.color("#009900");  });    socket.on('rojo', function (){  anode.on();  anode.color("#FF0000");  });    socket.on('stop', function (){  if (blink){  anode.stop(); *// to stop blinking*  blink = false;  }  else{  anode.blink(1000);  blink = true;  }  });    socket.on('off', function (){  anode.off(); *// to shut it off (stop doesn't mean "off")*  });    socket.on('on', function (){  anode.on(); *// to turn on, but not blink*  });    });  }); |

**Código HTML**

Con los métodos que proporciona Johnny-Five, actuaremos con el LED RGB. Ahora nos falta el código de nuestra interfaz web **(index.html)**.

|  |
| --- |
| <head>  <title></title>  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">  <style>  .btn{  width: 200px;  padding: 15px 32px;  text-align: center;  }  </style>  </head>  <body>  <div class="container theme-showcase" role="main">  <div class="jumbotron">  <h2>Firmata y Arduino</h1>  <p>Con este ejemplo, vamos a controlar un LED RGB a través de una web. Cuando pulsemos uno de los botones el LED, este cambiará al color que indique el botón. También podemos controlar encender o apagar el LED o que parpadee</p>  </div>  <div class="row">  <div class="col-md-3 col-md-offset-3">  <p> <button id="b\_azul" type="button" class="btn btn-lg btn-primary">Azul</button>  <p> <button id="b\_verde" type="button" class="btn btn-lg btn-success">Verde</button>  <p> <button id="b\_rojo" type="button" class="btn btn-lg btn-danger">Rojo</button>  </div>  <div class="col-md-3">  <p> <button id="b\_stop" type="button" class="btn btn-lg btn-default">Blink</button>  <p> <button id="b\_off" type="button" class="btn btn-lg btn-default">Led OFF</button>  <p> <button id="b\_on" type="button" class="btn btn-lg btn-default">Led ON</button>  </div>  </div>  <div class="row">  <div class="well">  <p> Realizado por:  <br> Extraído de <a href="https://programarfacil.com/blog/controla-tu-arduino-con-una-web-con-firmata-y-javascript">Programar Fácil</a>  </div>  </div>  </div>  <script src="https://netdna.bootstrapcdn.com/bootstrap/3.1.1/js/bootstrap.min.js"></script>  <script src="http://cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.min.js"></script>  <script src="app.js"></script>  <script>  var socket = io();    function cambiaAzul(){  socket.emit('azul');  }  function cambiaVerde(){  socket.emit('verde');  }  function cambiaRojo(){  socket.emit('rojo');  }    function stopBlink(){  socket.emit('stop');  }  function offLed(){  socket.emit('off');  }  function onLed(){  socket.emit('on');  }      document.getElementById('b\_azul').onclick = cambiaAzul;  document.getElementById('b\_verde').onclick = cambiaVerde;  document.getElementById('b\_rojo').onclick = cambiaRojo;  document.getElementById('b\_stop').onclick = stopBlink;  document.getElementById('b\_off').onclick = offLed;  document.getElementById('b\_on').onclick = onLed;  </script>  </body>  </html> |

De esta manera, tu aplicación IoT quedaría con la siguiente interfaz:

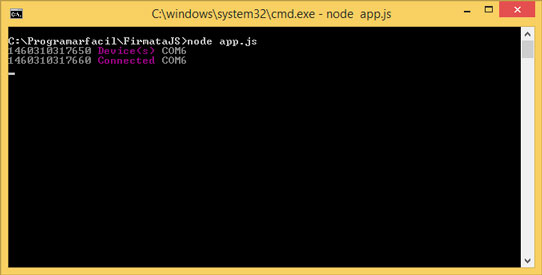


Fuente: [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/)

Teniendo tanto el archivo **app.js**, como **index.html** en nuestra carpeta raíz del proyecto, solo nos queda ejecutar la aplicación. En el terminal, lanzamos el comando que aparece a continuación

|  |
| --- |
| node app.js |

Esto iniciará el servidor y notificará que se ha conectado correctamente a la placa como aparece a continuación:



Fuente: [Programar Fácil](https://programarfacil.com/blog/arduino-blog/arduino-via-web-con-firmata/)

Por último solo debes revisar tu navegador web y lanzar la interfaz ingresando la siguiente dirección: localhost:8081. Desde esta interfaz podrás interactuar y controlar Arduino vía web gracias a Firmata y Javascript con el framework Johnny-Five.

Si bien, es un ejemplo sencillo, ahora les toca dar rienda suelta a su imaginación y controlar cualquier componente de su Arduino.