# Capítulo 7 – Comunicación NodeJS con Arduino

Este capítulo introduce sobre un framework denominado Johnny-five el cual es utilizado para la comunicación entre la aplicación web y las placas Arduinos que componen al SAR. Además, se explica en detalle la utilidad de un protocolo, denominado Firmata, que es cargado en cada uno de los Arduinos.

## **7.1 Johnny-five**

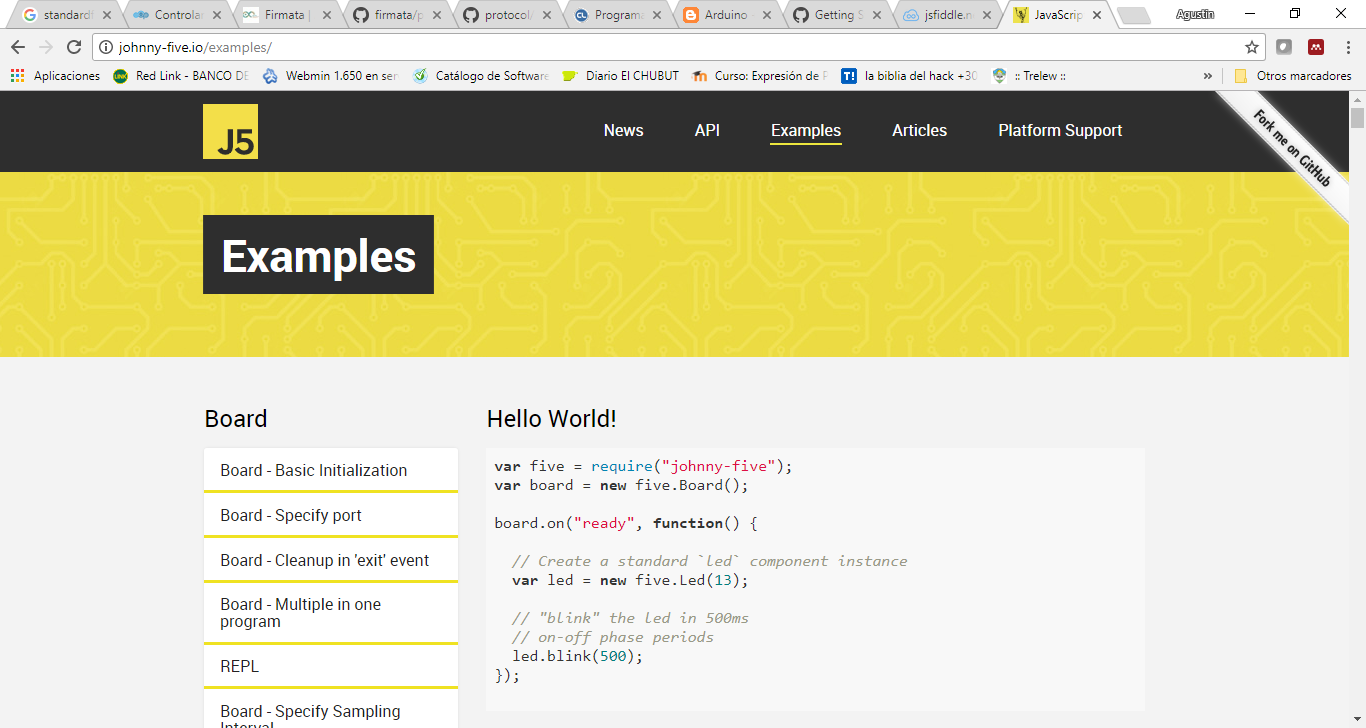
Johnny-five, más conocido como J5, es un framework de programación robótica basado en javascript lanzado por la compañía de tecnología web (Ilustración 42 - Página oficial de Johnny-Five[[1]](#footnote-1)) Bocoup en el 2012, es de código abierto, y cuenta con una gran variedad de desarrolladores, ingenieros y colaboradores que están constantemente perfeccionando y agregando características nuevas a esta herramienta.

Ilustración – Sitio web oficial de Johnny-Five

Es compatible con la mayoría de los microcontroladores y SBC más populares como, los utilizados en el desarrollo de esta tesina, Arduino y Raspberry Pi (ambos en todas sus versiones). Además de placas como BeagleBone, SparkFun, ChipKit, Intel-Galileo, entre otras.

Posee una librería muy completa, con compatibilidad a una vasta cantidad de sensores y actuadores, y ejemplos útiles de usos e implementación de los mismos. Dichos ejemplos se encuentran disponibles en su página oficial con el esquema de conexión de los componentes a las distintas plataformas compatibles.

## **7.2 Instalación**

Para poder utilizar Johnny-Five, se debe contar con Node.js (visto en el apartado **¡Error! No se encuentra el origen de la referencia.**).

La instalación de este módulo es sencilla, dado que la misma se puede agregar al paquete de librerías que se obtienen al crear un proyecto de Node.js. Para ello se debe ejecutar el siguiente comando, en una consola, dentro del directorio del proyecto:

npm install Johnny-five

Como se puede apreciar, se utiliza el comando npm que no es más que el gestor de módulos y aplicaciones de Node.js

## **7.3 Arduino Firmata**

Firmata es un protocolo genérico utilizado para la comunicación con microcontroladores desde software instalado en una computadora. Este protocolo se puede implementar en cualquier arquitectura de microcontroladores, así como en cualquier paquete de software.

Ilustración – Firmata como interfaz

El objetivo de firmata es permitir controlar completamente un microcontrolador de forma remota (Ilustración 43 - Remote Wiring de Windows), por ejemplo Arduino, desde un programa instalado en una computadora, sin escribir código de Arduino[[2]](#endnote-1).

Ventajas:

* Nuestro programa no está limitado por la memoria RAM y Flash de Arduino
* El software de control se puede programar en cualquier lenguaje, no solo C++, siempre que tenga soporte para firmata. Por ejemplo: Firmata: Processing, Visual Basic, Perl, C#, PHP, Java, **JavaScript**, Ruby y Python.

Desventajas:

* Programas más restringidos, es posible que algunas operaciones complejas no sea posible hacerlas. Por ejemplo, el uso de interrupciones.
* El microcontrolador no es autónomo, es decir, siempre debe estar conectado al computador para poder recibir comandos.

## **7.4 Surgimiento y funcionamiento de Firmata**

Como se mencionó en el apartado anterior, Firmata es un protocolo diseñado para la comunicación con microcontroladores desde cualquier software o computadora conectada al mismo. Su objetivo primordial es hacer del microcontrolador una extensión del entorno de desarrollo.

Fue diseñado para ser abierto, flexible y para que pueda ser soportado por cualquier lenguaje de programación, también debe ser simple de implementar en el microcontrolador y en el computador.

El protocolo Firmata fue creado en el año 2006 como una demo para Arduino por Hans-Chistoph Steiner. La idea de este protocolo se le ocurrió mientras estaba trabajando en un proyecto musical con dispositivos Eroktronix MIDItron conectados a varios Arduinos con varios sensores conectados. Cuando tenía que hacer una modificación en el código, debía replicarlos en todos los Arduinos.

La implementación de referencia es la librería para Arduino/Wiring la cual está incluida en el entorno de desarrollo oficial de Arduino a partir de la versión 0012.

Al trabajar con microcontroladores, siempre es necesaria la conexión del mismo con una computadora mediante comunicación serie. Estos son utilizados para la captura de datos mediante sensores y/o módulos conectados al mismo, y actuar en consecuencia según corresponda. Es por eso que, en este caso, surge la necesidad de un protocolo serie que permita la comunicación entre el microcontrolador y el computador. Debido a que este tipo de proyectos utilizan un único ambiente de programación, se hace evidente que los microcontroladores deberían ser una extensión del entorno de desarrollo en lugar de un dispositivo ajeno al proyecto, con su propio lenguaje de programación.

Inicialmente Firmata se centro en Arduino y en la actualidad sigue siendo su foco principal. La librería que cuenta con este protocolo fue portada a toda la familia Arduino y a Wiring. Por lo tanto, cualquier placa compatible puede utilizar Firmata o su librería asociada.

Firmata se basa en la API de Arduino (comandos del lenguaje) y los tipos de datos que se transfieren entre la placa y la computadora. Está enfocado en la facilidad de uso, aunque ello conlleve a un peor rendimiento.

Se le llama MIDI (Musical Instrument Digital Interface) a un protocolo, interfaz digital y conectores que se utilizan para la comunicación de computadoras, instrumentos musicales y otros dispositivos entre sí. Este protocolo fue el elegido como núcleo central de Firmata, debido a su eficiencia, facilidad y la cantidad de implementaciones existentes de acceso libre.

Firmata se basa en el formato de mensajes MIDI en el que los bytes de comando son 8 bits y los bytes de datos son 7 bits. Por ejemplo, el mensaje Midi Channel Pressure (Comando: 0xD0) tiene 2 bytes de longitud, en Firmata el comando 0xD0 se utiliza para habilitar informes para un puerto digital (colección de 8 pines). Las versiones MIDI y Firmata tienen 2 bytes de longitud, pero el significado es diferente. En Firmata, el número de bytes en un mensaje debe coincidir con el mensaje MIDI correspondiente. Los mensajes de Midi System Exclusive (Sysex) sin embargo, pueden tener cualquier longitud y, por lo tanto, se utilizan de forma prominente en todo el protocolo Firmata.

Firmata implementa solo el formato de mensajes de MIDI. El protocolo MIDI utiliza mensajes estándar para la comunicación. En el caso de Firmata estos mensajes han sido sustituidos y se creó un nuevo conjunto de mensajes tanto para los tipos de datos (analógicos y digitales) como para los mensajes de control (pinMode, digitalWrite, etc…). Esto hace posible representar la API de Arduino usando mensajes de Firmata, además, da soporte a más de 16 pines analógicos con una resolución de 14 bits y más de 128 pines digitales.

Por ser de código abierto, se han implementado diferentes versiones dando soporte a gran variedad de funcionalidades de los microcontroladores. La versión original, Standard\_Firmata, se incluye dentro de las versiones del entorno oficial de Arduino y Wiring e incluye soporte para las siguientes características[[3]](#endnote-2):

* Entradas y salidas analógicas
* Entradas analógicas
* Salidas PWM
* Conmutación entre entradas y salidas analógicas
* Servomotores
* Matrices de LEDs
* I2C

## **7.5 Métodos de librería Firmata en Arduino**

La librería Firmata de Arduino cuenta con un conjunto de métodos, relacionados con la API de Arduino (como se describió en el apartado anterior). A continuación, se detallan los más relevantes:

### 7.5.1 Métodos de propósito general

* **begin**(long): Comienza la librería, es posible utilizar otra velocidad diferente a la velocidad por defecto que es 57600 baudios. También es posible iniciar el protocolo Firmata desde otro Stream que no sea el que viene por defecto que es Serial.
* **printVersion()**: Envía la versión del protocolo al ordenador.
* **blinkVersion()**: Parpadea la versión de protocolo en el “build in LED”, generalmente el pin 13.
* **printFirmwareVersion():** Envía la versión de firmware y su versión al ordenador.
* **setFirmwareVersion(**byte major, byte minor**)**: Configura la versión del firmware.
* **setFirmwareNameAndVersion(**const char \*name, byte major, byte minor**)**: Configura nombre y versión del firmware.

### 7.5.2 Métodos para el envío de mensajes

* **sendAnalog(**byte pin, int value**)**: Envía el valor del pin analógico.
* **sendDigitalPort(**byte portNumber, int portData) : Envía el valor de un puerto digital de 8 bits.
* **sendString(**const char\* string**)**: Envía un string a una computadora.
* **sendString(**byte command, byte bytec, byte \*bytev**)**: Envía un string a la computadora usando un tipo de comando.
* **sendSysex(**byte command, byte bytec, byte\* bytev**)**: Envía un comando un con array de bytes
* **write(**byte c**)** – Envía un byte al stream de datos.

### 7.5.3 Métodos para la recepción de mensajes

* **available()**: Comprueba si hay algún mensaje entrante en el buffer.
* **processInput()**: Procesar los mensajes entrantes que hay en el buffer, mandado los datos a cualquiera de las funciones de callback registradas.
* **attach(**byte command, callbackFunction myFunction**)**: Registrar una función a un tipo de mensaje entrante.
* **detach(**byte command**)** : Suprimir la función del tipo de mensaje

### 7.5.4 Otros métodos

* **sendValueAsTwo7bitBytes(**int value) : Escribe el valor como 2 bytes.
* **startSysex(**void**)**: Comenzar mensaje sysex.
* **endSysex(**void**)**: Finalizar mensaje sysex[[4]](#endnote-3).

## **7.6 Instalación de Firmata en Arduino**

Con la instalación del IDE Arduino en una computadora se incluyen librerías y ejemplos que permiten manipular diversos componentes que se conecten a la plataforma. Dentro de estos ejemplos de códigos se encuentran los del protocolo Firmata. Para instalar Firmata se necesita tener conectada la placa Arduino a la computadora a través de un puerto USB. Luego de ello se debe seleccionar desde el IDE de Arduino (Ilustración 44 - IDE de Arduino) el código Firmata, dentro de su respectiva librería, según los dispositivos que se requieran comunicar con el mismo. Para ello se debe ir a Archivo 🡪 Ejemplos 🡪 Firmata.

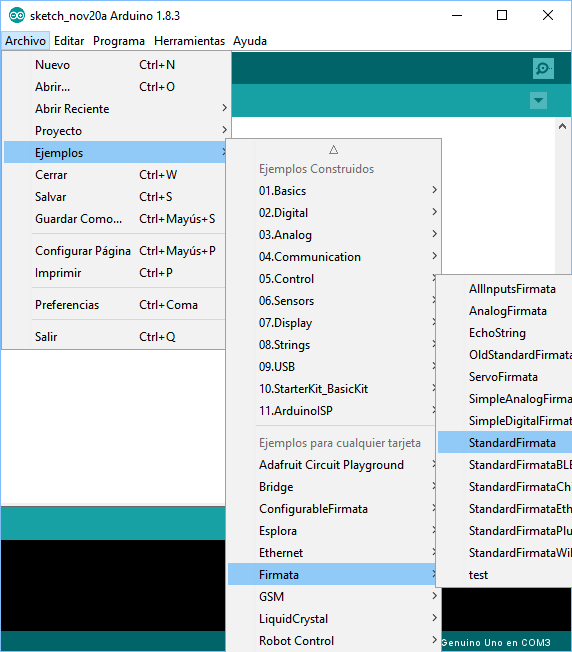


Ilustración - IDE de Arduino

En nuestro caso se utilizaron dos códigos Firmata:

**StandardFirmata**: Es, como su nombre lo indica, el estándar del protocolo que permite la comunicación con la mayoría de los componentes compatibles con Arduino. En nuestro caso, es el utilizado para cargarlo dentro del Arduino Mega para manipular la mayoría de sensores y actuadores del SAR. (Anexo de código)

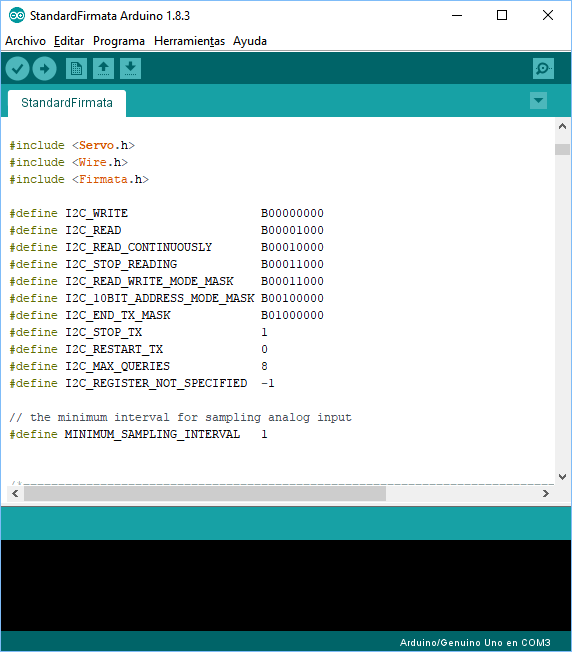


Ilustración - Código StandardFirmata

**ConfigurableFirmata**: Esta versión personalizada del protocolo[[5]](#footnote-2), es la utilizada para captar la temperatura mediante el sensor para dicho fin conectado al Arduino Nano. Permite separar las características del protocolo en clases individuales, haciendo más sencillo mezclar las características estándar del protocolo con otras personalizadas. (Anexo de código)

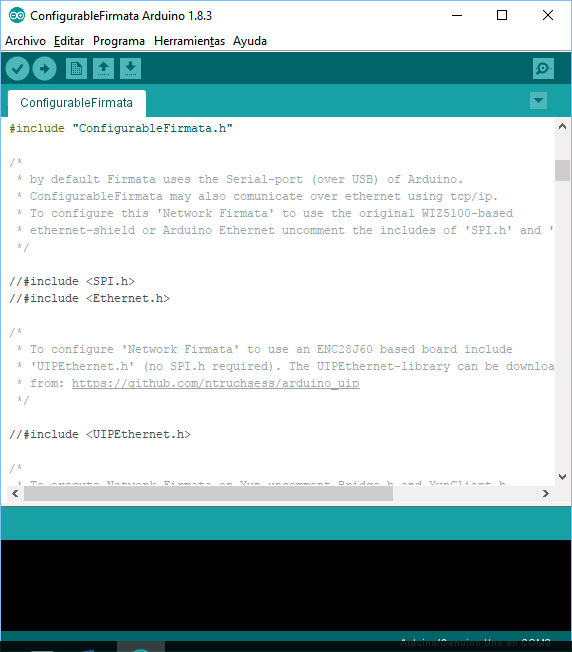


Ilustración - Código ConfigurableFirmata

## **Resumen**

Como se pudo ver, en este capítulo, se abordo la utilidad en el SAR de un framework denominado Johnny-five, el cual es utilizado para la programación robótica y está basado en javascript, además, se detalló el método de instalación del mismo para el entorno Node.js.

Por otro lado, se describió un protocolo genérico de comunicación entre microcontroladores y software de un computador, denominado Firmata. Detallando ventajas y desventajas en su utilización. Se explicó su surgimiento y su funcionamiento en general, como es el formato de sus mensajes. Finalmente se vio como instalar y/o cargar este protocolo en la familia de placas Arduino y los dos códigos utilizados en el desarrollo de esta tesina (StandardFirmata y ConfigurableFirmata)

Agregar los códigos StandardFirmata y ConfigurableFirmata (en Anexos de códigos)

1. Sitio oficial de Johnny-five: <http://johnny-five.io/> [↑](#footnote-ref-1)
2. <https://github.com/firmata/arduino> [↑](#endnote-ref-1)
3. <https://programarfacil.com/podcast/arduino-day-protocolo-de-comunicaciones-firmata/> [↑](#endnote-ref-2)
4. <https://www.arduino.cc/en/Reference/Firmata> [↑](#endnote-ref-3)
5. Se puede obtener del siguiente sitio web <https://github.com/firmata/ConfigurableFirmata> [↑](#footnote-ref-2)