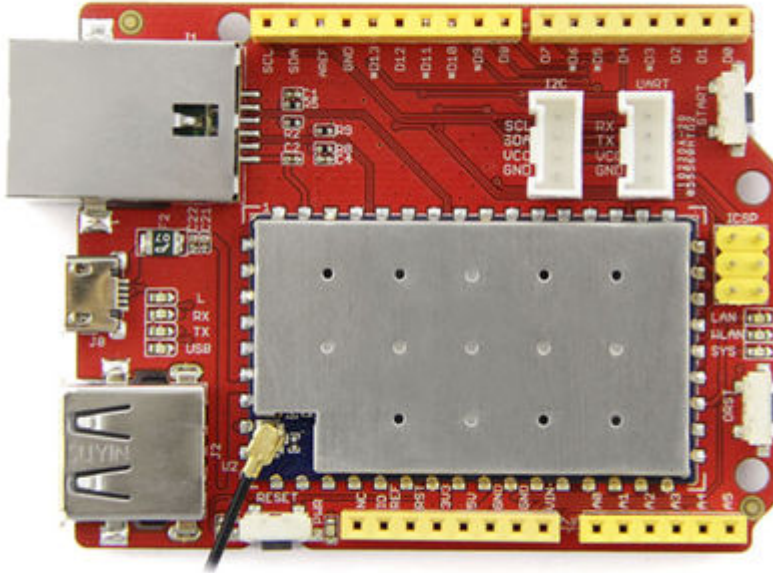


# Seeeduino Cloud – Parte 1



## Introducción

Recientemente he adquirido un Seeeduino Cloud. Un clon compatible con Yun. La verdad es que es muy versátil, fácil de configurar y potente. En este primer post hablaremos un poco de él, lo configuraremos y habilitaremos su Api Rest para encender y apagar el led que incorpora la placa.

- Configuración de red
- Configuración con WebGui
- Programa de ejemplo y test

## Configuración de red

Cuando arranquemos por primera vez nuestro Seeeduino, este levantará una wifi llamada **SeeeduinoCloud-AXXX** a la que nos podremos conectar. Una vez echo esto, podremos acceder a la configuración web a través de la ip **192.168.240.1**.



Welcome to your Seeeduno Cloud. Please enter password to access the web control panel

PASSWORD

Please be sure you have cookies enabled before proceeding.

LOG IN

El password por defecto es **seeeduno**.

### Configuración con WebGui

Después de entrar en la web, la interfaz nos mostrará el estatus de las redes WiFi/ Eth. En la parte superior derecha encontraremos las siguientes opciones:

- SYSTEM -> Configuración global
- SENSORS -> configuración del servidor IoT
- UPGRADE -> Actualización de firmware



WELCOME TO SEEDUINO CLOUD

SYSTEM

SENSORS

UPGRADE

Seleccionaremos SYSTEM y configuraremos nuestro Seeeduno Cloud:

- Primeramente podemos modificar el password de acceso
- A continuación seleccionaremos la red wifi a la que queremos conectar nuestro Seeeduno Cloud y estableceremos el password de la misma
- Finalmente, podemos proteger nuestra red con una password para la Api Rest. Si lo hacemos debemos tener

en cuenta que será una protección Basic Auth y el usuario por defecto será root

For more advanced network configuration features, see the [advanced configuration panel \(luci\)](#)

### CLOUD CONFIGURATION

SEEDUINO CLOUD NAME \* Seeed

PASSWORD

CONFIRM PASSWORD

TIMEZONE \* Rest of the World (UTC)

### WIRELESS PARAMETERS

CONFIGURE A WIRELESS NETWORK

DETECTED WIRELESS NETWORKS Select a wifi network... [Refresh](#)

WIRELESS NAME \*

SECURITY WPA2

PASSWORD \*

DISCARD CONFIGURE & RESTART

### REST API ACCESS

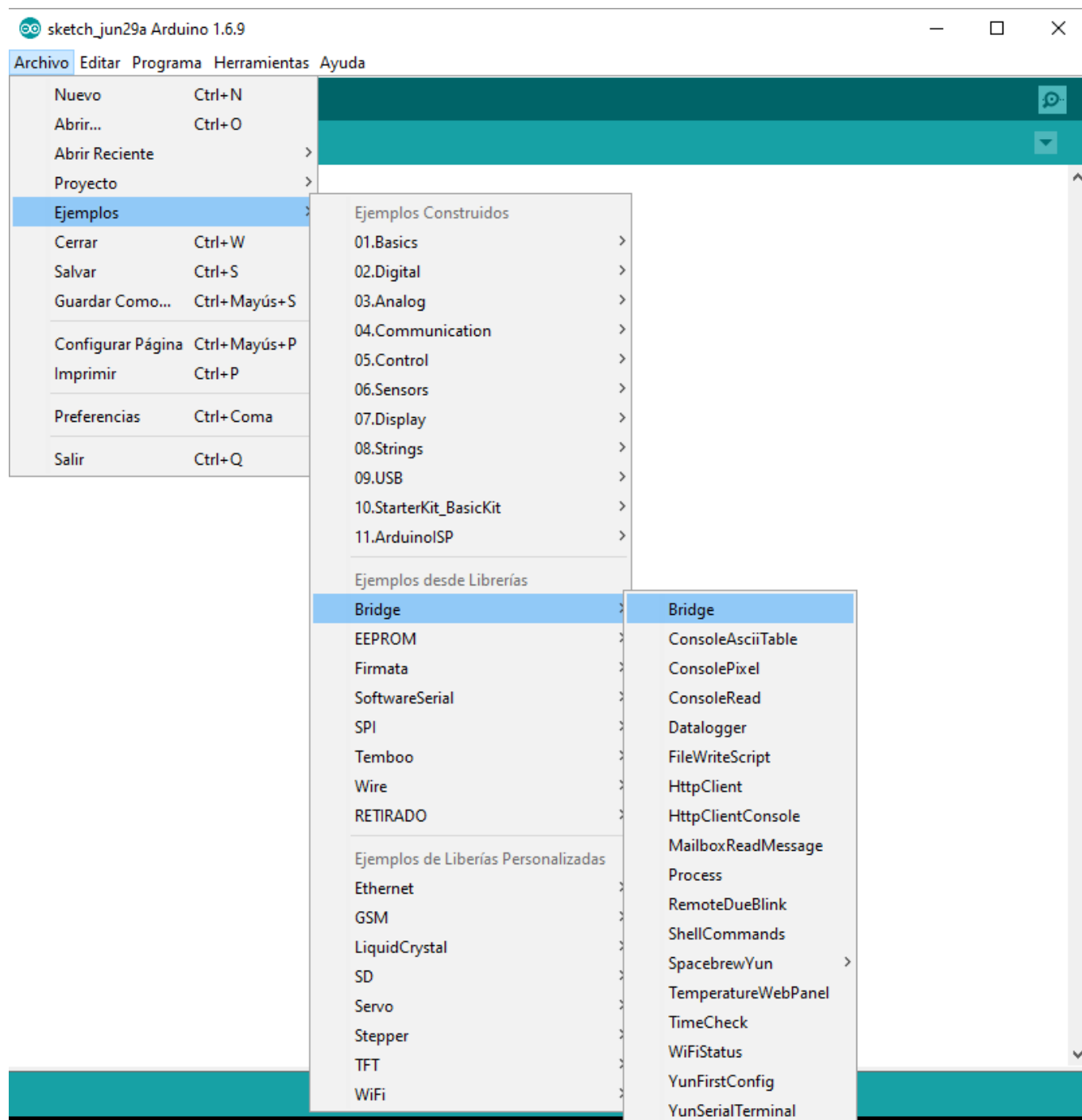
REST API ACCESS  OPEN  WITH PASSWORD

REST APIs allow you to access your sketch from the web, sending commands or exchanging configuration values. If your Yún is on a public network, or controlling sensitive equipment, or both, we recommend you leave the REST API password protected.

Cuando pulsemos en **CONFIGURE & RESTART** nuestro Seeeduino se configurará, se reiniciará y se conectará a nuestra red.

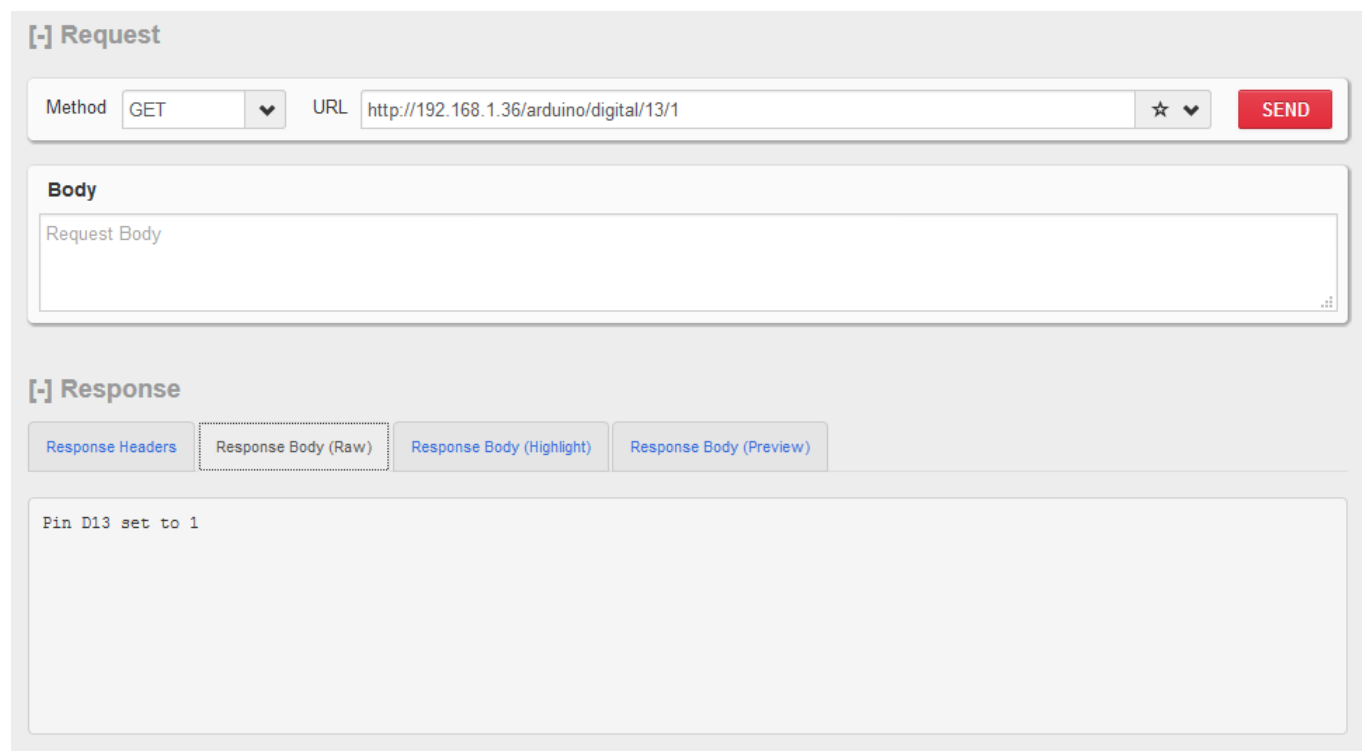
Programa de ejemplo y test

Bien lo siguiente sera subir un código de ejemplo al Arduino para poder testear la aplicación. Para más información sobre como subir un programa des del IDE a nuestro Arduino podéis mirar este post [Arduino – Primera Parte](#). Utilizaremos un ejemplo de la librería **Bridge**.



Una vez subido nuestro programa atacaremos la Api Rest para encender y apagar el led que vine incorporado en la placa, exactamente el 13 (situado justo detrás del puerto microusb). En esta caso utilizo el plugin para Firefox [RESTclient](#), aunque

podríamos utilizar cualquier otro. En la url especificamos la ip de nuestro Seeeduino, arduino(podemos acceder a otros elementos de la placa), el tipo de pin (en este caso digital), el pin (13) y el valor (0 apagado, 1 encendido). La url final sería **http://192.168.1.36/arduino/digital/13/1** (encendido) o **http://192.168.1.36/arduino/digital/13/0** (apagado)



The screenshot shows a REST client interface with two main sections: Request and Response.

**Request Section:**

- Method: GET
- URL: `http://192.168.1.36/arduino/digital/13/1`
- Buttons: A star icon and a red "SEND" button.
- Body: A text area labeled "Request Body" which is currently empty.

**Response Section:**

- Buttons: "Response Headers", "Response Body (Raw)", "Response Body (Highlight)", and "Response Body (Preview)".
- Response Body: A text area containing the text "Pin D13 set to 1".

## **Conclusión**

Como podemos ver la potencia de Seeeduino Cloud es mucha. Fácilmente podemos acceder a nuestro Arduino recibiendo y enviando valores a través de una sencilla Rest Api. En el próximo post acoplaremos un Relay Shield, esto nos permitirá controlar remotamente aparatos electrónicos (dado el calor que hace ahora, un pequeño ventilador)