

# Node-RED

Node-RED una herramienta de programación para conectar dispositivos de hardware, APIs y servicios en línea. Provee un **editor web** que permite crear flujos usando nodos de una paleta que se conectan con *cables*. Permite **desplegar** el proyecto con un solo click.

El editor está basado en una paleta, uno o más flujos, y un botón de despliegue.

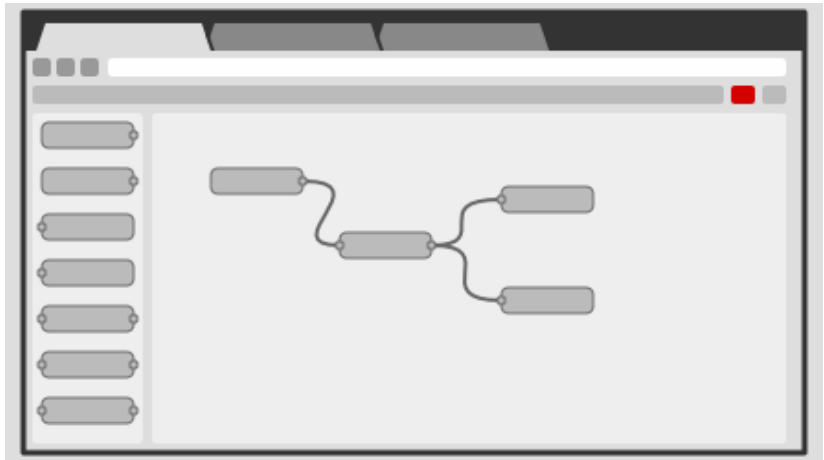


Figura1: Node RED

# Características

- Un proyecto es simplemente un archivo **JSON** que puede compartirse por cualquier método.

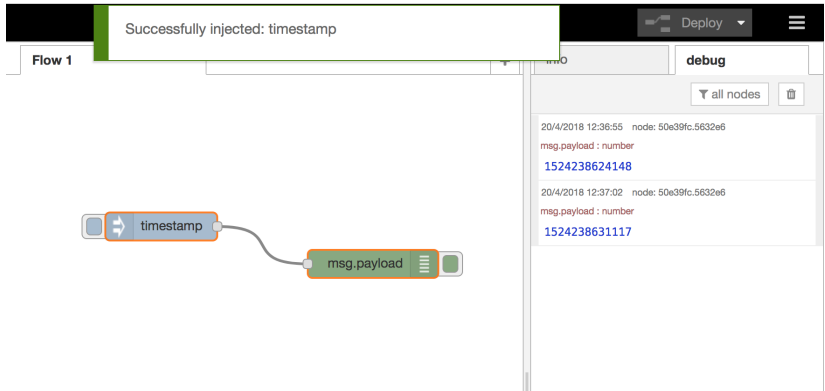


Figura2: Inyección de fecha a nodo de depuración

Su definición de flujo es la siguiente:

```
[  
  {  
    "id": "278c9650.83264a",  
    "type": "tab",  
    "label": "Flow 1",  
    "disabled": false,  
    "info": ""  
  },  
]
```

```
{  
  "id": "ca9bd2eb.a07ab",  
  "type": "inject",  
  "z": "278c9650.83264a",  
  "name": "",  
  "topic": "",  
  "payload": "",  
  "payloadType": "date",  
  "repeat": "", "crontab": "",  
  "once": false,  
  "onceDelay": 0.1,  
  "x": 175.5, "y": 580,  
  "wires": [["50e39fc.5632e6"]]  
},
```

```
{  
  "id": "50e39fc.5632e6",  
  "type": "debug",  
  "z": "278c9650.83264a",  
  "name": "",  
  "active": true,  
  "tosidebar": true,  
  "console": false,  
  "tostatus": false,  
  "complete": "false",  
  "x": 402.5,  
  "y": 620,  
  "wires": []  
}  
]
```

## Características (cont.)

- ▶ Soporte para **entradas** y salidas de HTTP, WebSockets, MQTT, RSS, Mail, TCP, UDP, captura de excepciones.
- ▶ Nodos que ejecutan alguna función:
- ▶ Control de flujo
- ▶ Generación de texto con plantillas.
- ▶ *Parseo* y manipulación de cadenas.

# Características (cont.)

## Funciones definidas por el usuario

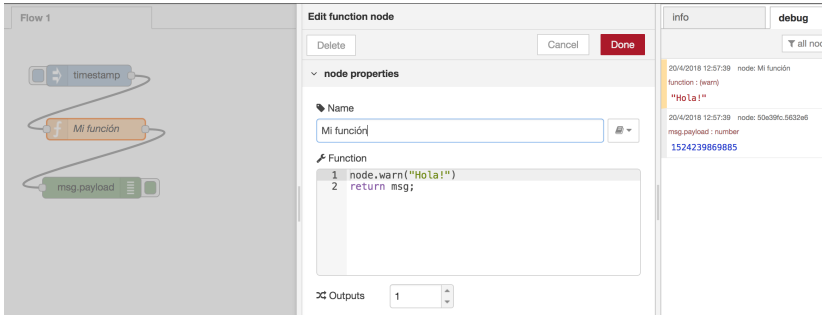


Figura3: Funciones definidas por el usuario



## Características (cont.)

- Gran cantidad de **plugins de terceros** instalables desde la misma interfaz web.

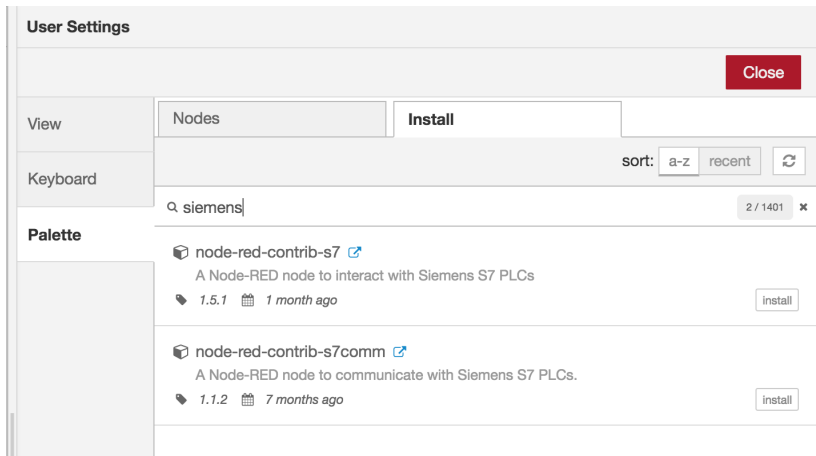


Figura4: Nodos de Terceros

## Características (cont.)

- ▶ Se ejecuta en dispositivos reducidos como Raspberry Pi, Beagle Bone y derivados, permitiendo computación de borde (*edge computing*).
- ▶ Soporte para varias nubes como *IBM Bluemix*, *SenseTecnica FRED*, *Amazon Web Services* o *Microsoft Azure*.
- ▶ Permite la adaptación de la interfase (restricción de nodos, cambio de estilo, etc.).
- ▶ Soporte para mecanismos de autenticación como HTTP Basic o OAuth2.
- ▶ Plugins para generación de ***dashboards***.
- ▶ Gestión de proyectos con DVCS
- ▶ Integración con frameworks como *Express*

## Ejecución y directorio de usuario

Cuando Node-RED se inicia, presenta por salida estándar mensajes del estilo:

```
Welcome to Node-RED
=====
```

```
Node-RED version: v0.18.4
Node.js version: v6.13.1
Linux 4.9.87-linuxkit-aufs x64 LE
Loading palette nodes
```

```
-----
[node-red/rpi-gpio] Info : Ignoring Raspberry Pi specific m
-----
```

```
Settings file   : /data/settings.js
User directory  : /data
```

Flows file : /data/flows.json

Creating new flow file

Server now running at http://127.0.0.1:1880/

y crea algunos archivos en el directorio dónde se lanzó el proceso o el especificado por `-u` o `-userDir`.

- ▶ settings.js
- ▶ flows.js
- ▶ package.json

## Ejecución y directorio de usuario (o proyecto)

- ▶ El archivo `settings.js` es creado en la primera ejecución.
- ▶ El archivo `flows.js` se actualiza con cada **deploy**, pero no es creado inicialmente.
- ▶ `package.json` se crea solo si se activa la gestión de proyectos
- ▶ Puede existir un directorio de paquetes de node si se instalan módulos desde el menú ***Manage Palette*** del menú hamburguesa de la esquina superior.

## Definición de funciones

El nodo **function** almacena código que se ejecuta ante la llegada de un mensaje. La función se envuelve automáticamente por Node-RED, por lo que no hace falta notación de función de JavaScript.

```
function (msg, node, context) {  
    // Código introducido en el editor  
    return msg  
    // Fin de código introducido en el editor  
}
```

# Definición de funciones (cont.)

## Múltiples salidas

Si una función tiene múltiples salidas, se debe retornar un arreglo con `null` en cada salida no activada.

```
if (msg.topic === "banana") {  
    return [ null, msg ];  
} else {  
    return [ msg, null ];  
}
```

Puede activarse varias salidas al mismo tiempo y enviar inclusive el mismo mensaje.

## ▼ node properties

 Name

Multisalida



 Function

```
1  
2 return [{  
3     payload: 1  
4 }],  
5 {  
6     payload: 2  
7 }, {  
8     payload: 3  
9 }];
```

 Outputs

3

See the Info tab for help writing functions.

Figura5: Múltiples salidas



# Definición de funciones (cont.)

## Múltiples mensajes

Se pueden emitir múltiples mensajes por una salida, para esto se debe generar un arreglo con los valores a retornar:

```
var msg1 = { payload:"first out of output 1" };
var msg2 = { payload:"second out of output 1" };
var msg3 = { payload:"third out of output 1" };
var msg4 = { payload:"only message from output 2" };
return [ [ msg1, msg2, msg3 ], msg4 ];
```

## Definición de funciones (cont.)

### Tareas asíncronas

Si un nodo requiere realizar tareas asíncronas, el objeto node provee el método send.

```
doSomeAsyncWork(msg, function(result) {  
    node.send({payload:result});  
});  
return;
```

```
// Si hubiera tareas de limpieza al finalizar,  
// también se puede usar el evento `close`  
node.on('close', function() {  
    // Limpieza, cancelaciones, cierres  
});
```

# Definición de funciones (cont.)

## Logging y Errores

Para depuración es posible utilizar otros atributos de node:

```
node.log("Información");  
node.warn("Algo que el usuario debería enterarse");  
node.error("Oh no, esto es grave");
```

Si se utiliza `node.error("Mensaje", msg)` el error puede ser capturado por el nodo `catch` del mismo flujo.

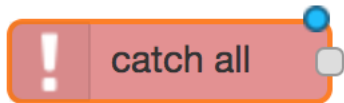


Figura6: Captura de errores

## Definición de funciones (cont.)

### Almacenamiento

Además de los mensajes un nodo puede tener un contexto donde almacenar información durante su ejecución.

```
// Inicialización  
var count = context.get('count') || 0;  
count += 1;  
// Almacenamiento  
context.set('count', count);  
// enviando el valor como parte de un mensaje  
msg.count = count;
```

Para compartir datos entre nodos de un mismo flujo se puede usar de manera similar `flow.set` y `flow.get`, y de manera global `global.get` y `global.set`.

## Definición de funciones (cont.)

### Cambio de estado del nodo

Se puede alterar la presentación del nodo en el flujo de la siguiente manera:

```
node.status({fill:"red",shape:"ring",text:"disconnected"});  
node.status({fill:"green",shape:"dot",text:"connected"});  
node.status({text:"Solo estado"});  
node.status({});    // Limpiar estado
```



Figura7: Estado de los nodos

Más información en  
<https://nodered.org/docs/creating-nodes/status>

# Definición de funciones (cont.)

## Módulos de NodeJS comunmente usados

- ▶ Buffer para manejo de datos binarios.
- ▶ `setTimeout/clearTimeout` y `setInterval/clearInterval` para temporización.
- ▶ `util` para verificación de tipos, conversión de *callbacks* a *Promise*, formateo de cadenas ala *printf*, etc.

# Documentación

## Documentation

---



### Getting Started

Everything from first install to deploying flows



### User Guide

The definitive guide to using Node-RED



### Cookbook

Recipes to help you get things done with Node-RED



### Creating Nodes

How to create nodes to extend the Node-RED palette



### Developing the core

Help to develop the core of Node-RED



### API Reference

Admin, runtime and storage APIs

Figura8: <https://nodered.org/docs/>



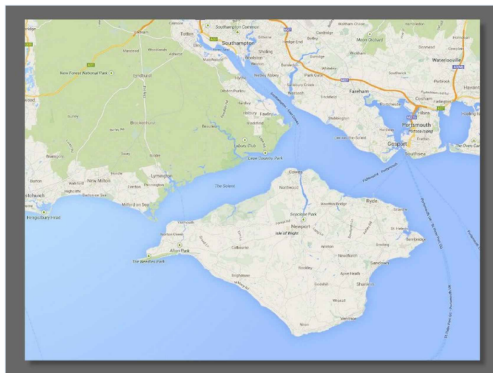
# Presentación oficial



Download [MP3](#) | [Slides](#) | [Android app](#) 39:26

## Summary

Nick O'Leary introduces Node-RED, an open source and browser-based environment for visually wiring together the many different streams that exist in the world of IoT.



## Bio

Nick O'Leary is an Emerging Technology Specialist at IBM where he gets to do interesting things with interesting technologies and also play with toys. With a background in pervasive messaging, he is a developer on the Eclipse Paho project and sits on the OASIS MQTT Technical Committee. He is the creator of Node-RED, an open source tool for wiring the Internet of Things.

Figura9: Creador de Node-RED presentándolo en QCon London 2014 <sup>2</sup>[<sup>2</sup>]

<sup>2</sup><https://www.infoq.com/presentations/ibm-node-red>

Fin!