CONOSCERE E USARE

Node-RED

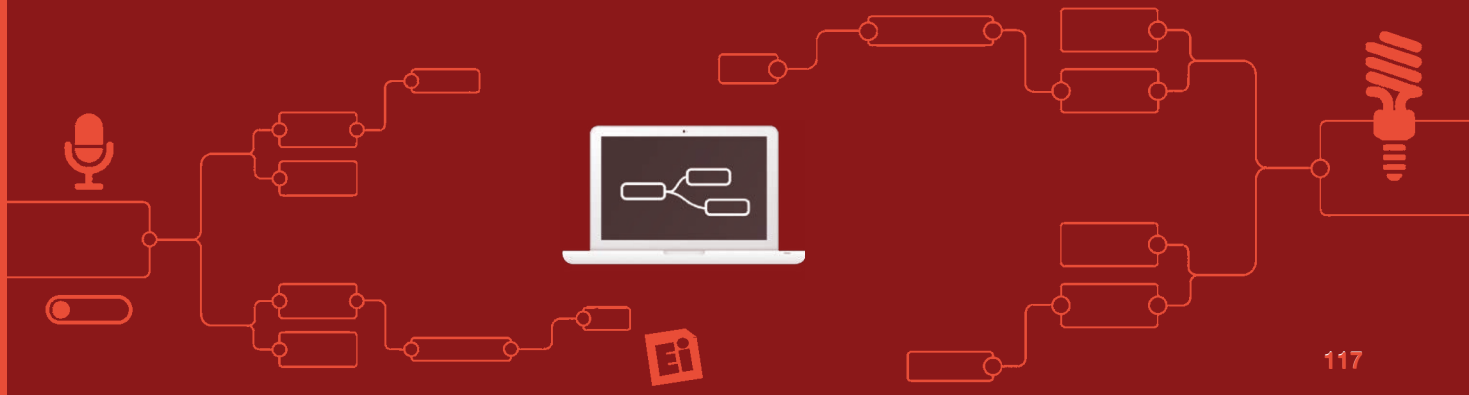
di FRANCESCO FICILI

1

Iniziamo a scoprire un tool di flow-based programming, orientato all'IoT ed alla connettività, originariamente sviluppato dall' IBM Emerging Technology Services team e adesso parte della JS Foundation. Spiegheremo cos'è Node-RED ed inizieremo a prendere confidenza con l'ambiente e con l'editor grafico, mostrando poi alcuni esempi di utilizzo su Raspberry PI. Prima Puntata.

Negli ultimi anni si è assistito ad una sempre maggiore diffusione di applicazioni IoT, tanto che ormai la connettività è una caratteristica di base della maggior parte degli oggetti che utilizziamo ogni giorno. Basti pensare ad un comune elettrodomestico moderno, come una lavatrice o un frigorifero, per renderci conto che ormai il fatto che l'oggetto sia connesso è una caratteristica quasi imprescindibile in ogni genere di applicazioni, per non parlare di oggetti più specializzati, come ad esempio gli assistenti vocali di Amazon e Google, che fanno della connettività e della loro vocazione IoT la loro caratteristica principale. Anche nel mondo dei maker e degli hobbisti si percepisce sempre più la necessità di poter connettere i propri progetti con la Rete, per eseguire le più svariate funzionalità; pensiamo ad esempio ad un impianto di irrigazione che desideriamo controllare dal nostro smartphone o ad una stazione meteo che ci fornisca i dati acquisti dai suoi sensori direttamente su un PC o via mail, o, ancora meglio, tramite Telegram.

Utilizzando piattaforme ed ambienti di sviluppo tradizionali è sicuramente possibile ottenere risultati di questo tipo, ma è necessario scrivere parecchie righe di codice, utilizzando librerie di una certa complessità che spesso pregiudicano la manutenibilità e la scalabilità delle nostre applicazioni. Una soluzione alternativa al problema, che semplifica moltissimo l'approccio e consente di avere delle applicazioni sempre facilmente leggibili e scalabili, è l'utilizzo di un ambiente grafico specializzato nella realizzazione di applicazioni IoT ed orientate alla connettività. Dato che questa necessità è fortemente sentita, un gruppo di programmatori della IBM ha sviluppato un ambiente di programmazione che segue questo approccio, dando vita a quello che oggi è conosciuto come Node-RED.



COS'È NODE-RED

Node-RED è un tool di programazione grafica sviluppato dall'Emerging Technology Services team di IBM.
Il progetto nasce all'inizio del 2013, da Nick O'Leary e Dave Conway-Jones, come un proof-of-concepts per la visualizzazione ed il mapping tra topics MQTT, per diventare velocemente un tool più generale che consente lo sviluppo di applicazioni di vario tipo. Il progetto è open-source praticamente dalla sua concezione, e quest'orientamento open è culminato nel divenire uno dei 32 progetti della JS fundation, nell'ottobre 2016.
Node-RED consiste sostanzialmente in una runtime di *Node.js*, che implementa un server al quale si può far puntare un qualsiasi browser per accedere all'editor grafico.
All'interno dell'editor visualizzato dal browser è possibile creare i programmi sotto forma di insieme di nodi (flussi o flows) interconnessi tra di loro.
I nodi vengono inseriti nei flussi prelevandoli da un'apposita palette presente nell'editor. La palette dei nodi può facilmente essere espansa tramite altri nodi creati dall'utente stesso o dalla community.
Una volta che il flusso è completo è possibile eseguirne il deploy e quindi mandarlo in esecuzione, tramite un apposito pulsante.
Essendo basato su Node.js, Node-RED può virtualmente essere eseguito su qualsiasi sistema in grado di supportare Node.js, compresi:

- PC Linux o Windows;
- sistemi embedded come Rasperry Pi o BeagleBoard;
- sistemi Android;
- server.

Nel resto di questo corso utilizzeremo Node-RED su Raspberry PI, ma i concetti esposti restano validi per l'utilizzo di Node-RED anche su altre piattaforme (sono riassunte nella Fig. 2); ciò che può cambiare sono principalmente alcune caratteristiche relative all'installazione, per le quali rimandiamo i lettori all'apposita sezione sul sito web ufficiale: *https://nodered.org/docs/getting-started/*.

FLOW-BASED PROGRAMMING

Il concetto principale su cui si basa Node-RED è quello della cosiddetta "Flow-Based Programming" o FBP. Tale paradigma di programmazione, enunciato per la prima volta da J. Paul Morrison nel 1970, descrive le applicazioni come una network di processi "black box", ossia di cui non si conoscono i dettagli implementativi, ma di cui sono noti i dati in ingresso, il tipo di elaborazione eseguita su di essi e i dati in uscita. I processi black box sono anche noti come nodi (questo è proprio il termine utilizzato da Node-RED) e tali nodi

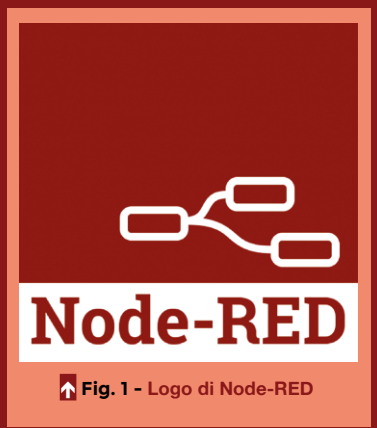


Fig. 1 - Logo di Node-RED

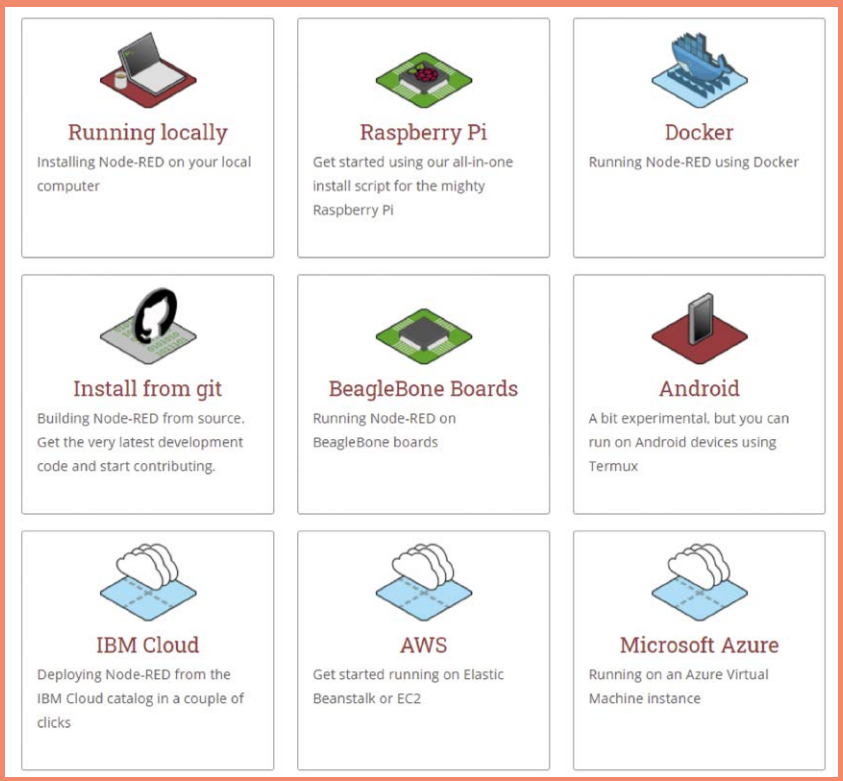


Fig. 2 - Le varie piattaforme che supportano l'esecuzione di Node-RED.

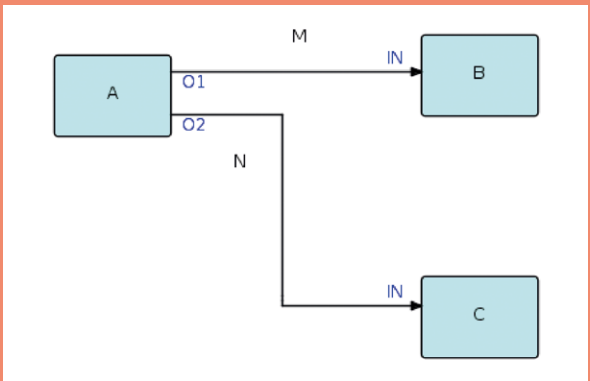


Fig. 3 - Un semplice diagramma di Flow-Based Programming.

si scambiano dati attraverso messaggi instradati tramite connessioni predefinite (network). Ne consegue che la FBP è naturalmente "component oriented", ossia è un paradigma che enfatizza la modularità nello sviluppo software.
La FBP vede un'applicazione non come un processo sequenziale, che inizia ad un certo istante di tempo e fa una cosa alla volta finchè non completa la sua esecuzione, ma piuttosto come un insieme di processi asincroni che comunicano tra di loro per mezzo di uno stream di dati strutturati, chiamati "information packets" (IPs).
Il focus è quindi sui dati applicativi e sulle trasformazioni applicate su di essi al fine di produrre l'output desiderato.
La network è definita esternamente ai processi e viene interpretata a sua volta come una parte di software, normalmente chiamata "scheduler".
In FBP i processi comunicano per mezzo di connessioni e tali connessioni sono collegate ai processi per mezzo di porte (in ingresso o in uscita); in un certo lasso temporale, un IP può essere manipolato solo da un singolo processo, oppure può essere in transito tra due processi.
Data la sua natura, la definizione di una network in FBP è normalmente un diagramma, ma può anche essere convertita in una lista di connessioni e nodi, se viene descritta con un linguaggio di basso livello o una particolare notazione (vedremo che questo avviene anche in Node-RED).
In Fig. 3 è riportato un semplice diagramma FBP.
A, B e C sono processi, mentre O1, O2 e IN sono porte. M ed N invece sono connessioni che connettono le porte dei processi tra di loro. Le connessioni hanno una capacità fissa in termini di numero di IPs che possono essere immagazzinati in un certo istante di tempo.
Node-RED implementa tutti i concetti base della FBP, quindi un programma o, per usare la terminologia Node-RED, un flow, è descritto come una serie di nodi interconnessi tra di loro. Ogni nodo implementa una specifica funzione, riceve un set di dati da una porta in ingresso, li manipola e li passa

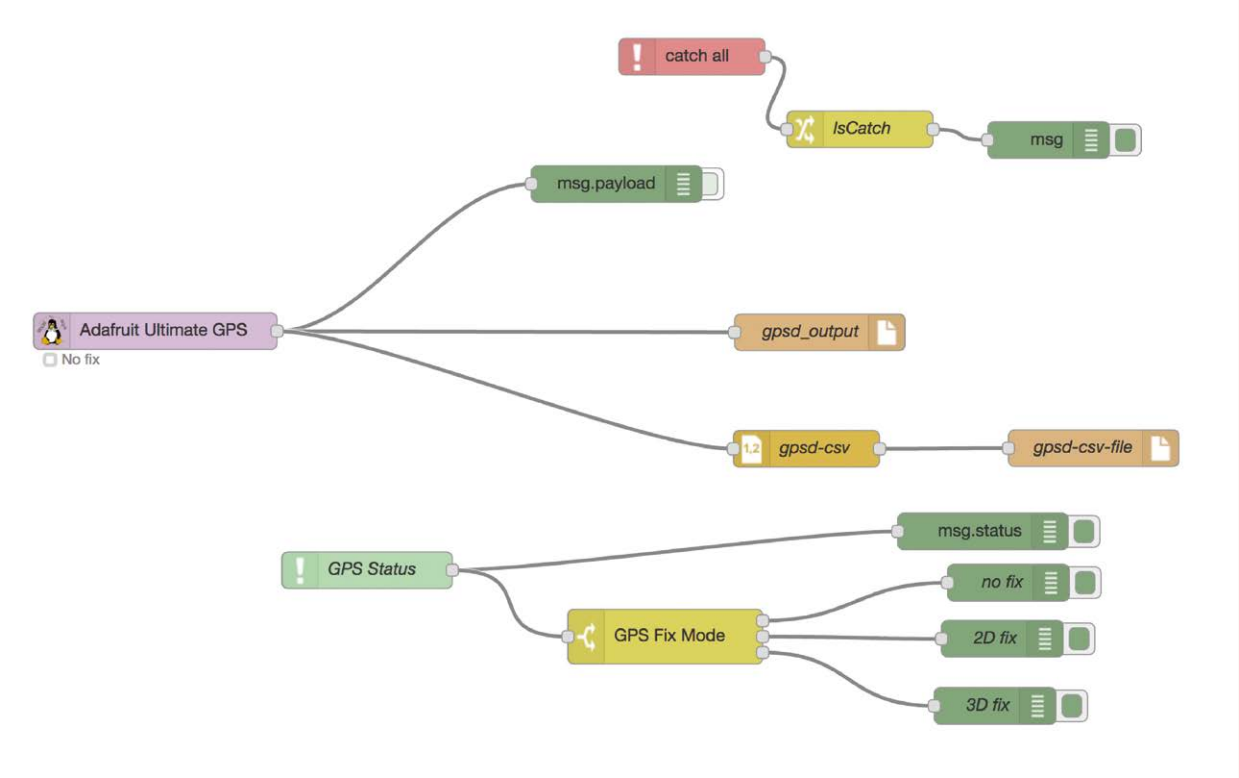


Fig. 4 - Esempio di Flow in Node-RED.

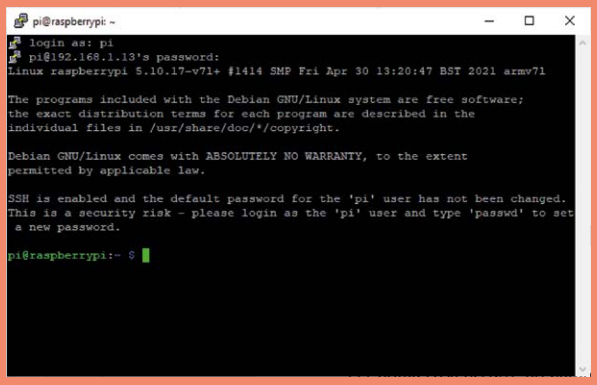


Fig. 5 - Schermata della console di Raspberry PI Os visualizzata in SSH tramite PuTTY.

alla porta/e in uscita. La network è responsabile del flusso di dati attraverso i vari nodi. In Fig. 4 è riportato un esempio di flow in Node-RED.
Prima di addentrarci sui dettagli della programmazione di un flow vediamo però come installare ed avviare l'ambiente.

INSTALLARE ED AVVIARE NODE-RED

Come abbiamo già accennato, Node-RED può girare su una moltitudine di piattaforme: di fatto, su ogni piattaforma capace di far girare Node.js. Dal punto di vista dell'utilizzo non c'è una grossa differenza, ma ovviamente la procedura di installazione può differire da caso a caso.
Oltre a questo in alcuni casi occorre preparare opportunamente la piattaforma. In questa sezione forniremo una serie di istruzioni complete per l'installazione su Raspberry Pi, ma si tenga conto che non si tratta di istruzioni valide in generale, quindi consultate la relativa sezione del sito web di Node-RED per maggiori informazioni.
Per installare ed eseguire Node-RED su Raspberry PI occorre innanzitutto munirsi di una Raspberry. Non ci sono particolari limitazioni rispetto al modello da utilizzare, ma si consiglia di non usare un modello troppo antiquato.
Per le nostre prove abbiamo utilizzato una Raspberry Pi 4 Model B con 4 GB di RAM.
Per prima cosa occorre preparare una scheda SD con un'immagine di Raspberry OS al suo interno, usando Raspberry Pi Imager. Se si utilizza un'immagine "full with recommended SW", Node-RED risulterà già preinstallato, ma sconsigliamo di utilizzare questa opzione sia per i tempi di creazione dell'immagine che per il fatto che la versione preinstallata di Node-RED potrebbe essere non aggiornata all'ultima versione.
In alternativa è possibile creare un'immagine di tipo Lite ed installare Node-RED successivamente.
Creiamo quindi un'immagine di tipo Lite e predisponiamo la Raspberry al collegamento con la nostra rete domestica in modalità headless, abilitando al contempo la connessione SSH. Per l'accesso headless rimandiamo alla guida reperibile su *https://www.raspberrypi.org/documentation/configuration/wireless/headless.md* presente sul sito web della raspberry PI fundation. Per abilitare la connessione SSH inserire un file privo di estensione e con nome "SSH" all'interno della partizione di boot della SD-card. Una volta eseguite queste operazioni sarà possibile collegarsi alla Raspberry Pi tramite un client SSH, come ad esempio PuTTY (Fig. 5).
Per installare Node-RED e le sue dipendenze principali (ossia Node.js e npm) è possibile utilizzare il seguente script, che installa i vari componenti ed i nodi specifici per Raspberry Pi. Lo script può anche essere utilizzato per aggiornare un'installazione di Node-RED esistente.

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/
linux-installers/master/deb/update-nodejs-and-nodered)
```

Più in dettaglio lo script esegue le seguenti operazioni:

- rimuove le versioni di Node-RED e Node.js preesistenti, se presenti;
- installa la versione di Node.js LTS corrente usando NodeSource; se Node.js è già presente viene eseguito un controllo di compatibilità per assicurarsi che Node sia almeno alla versione 12;
- installa l'ultima versione di Node-RED usando npm;
- se desiderato dall'utente installa una serie di nodi specifici per Raspberry Pi;
- setta Node-RED per funzionare come service e fornisce una serie di comandi specifici per utilizzare i servizi.

Il risultato dell'esecuzione è quello proposto nella schermata di Fig. 6. Si noti inoltre che Node-RED è stato anche inserito nei repository di Raspberry Pi Os, e compare nella lista dei software raccomandati dalla foundation, e può quindi essere installato anche tramite apt-get (apt-get

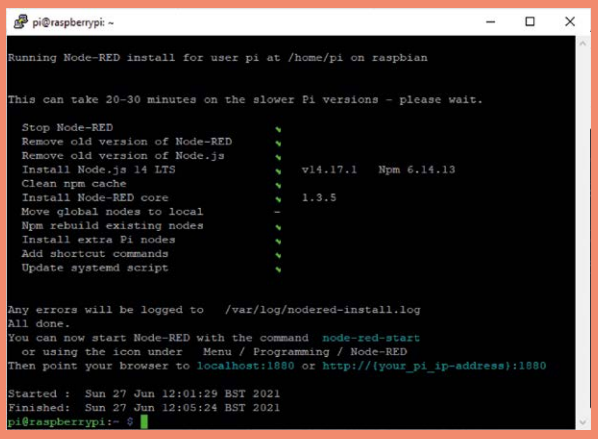


Fig. 6 - Schermata della console dopo l'esecuzione dello script

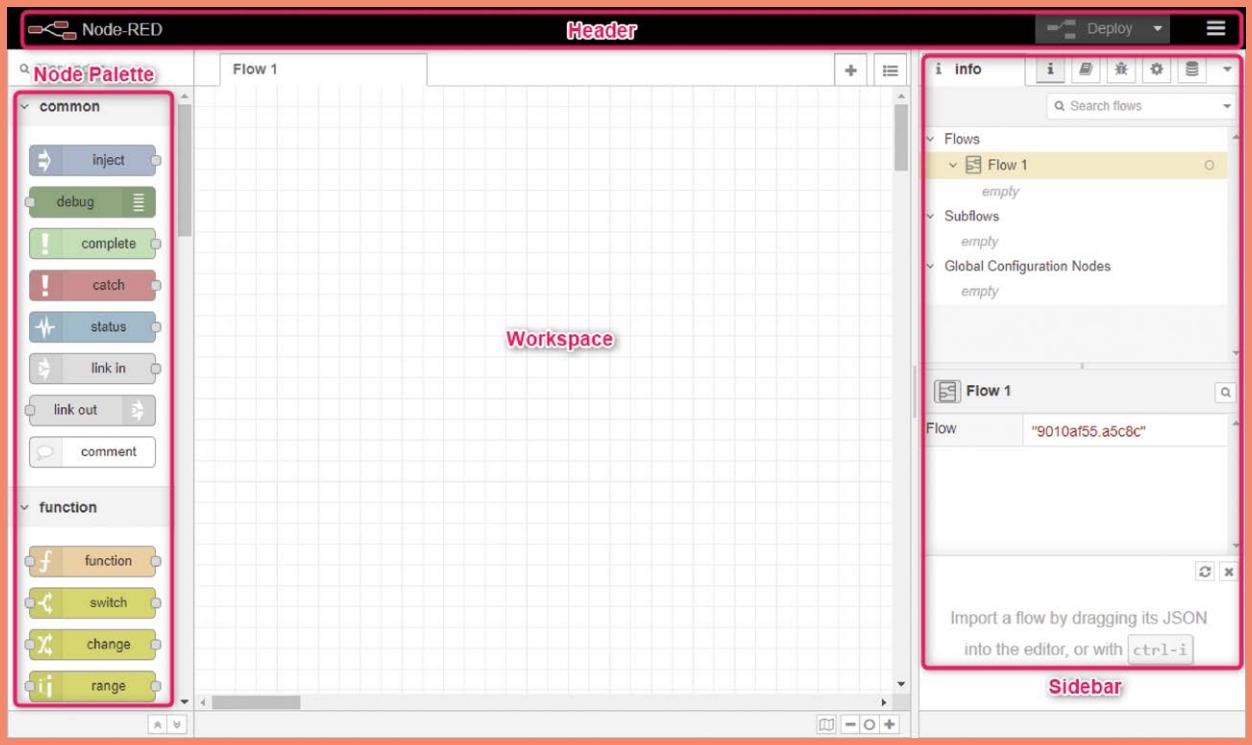


Fig. 7 - Editor di Node-RED.

install nodered). Tuttavia, con questa procedura non verrà installato npm (mentre è comunque incluso Node.js).
La documentazione di Node-RED comunque suggerisce l'utilizzo dello script per installare ed aggiornare Node-RED su Raspberry PI. Come spiegato, lo script esegue anche i settaggi necessari per permettere la gestione di Node-RED come servizio; ciò significa che è possibile far girare Node-RED in background ed avviarlo automaticamente al boot. Sono disponibili i seguenti comandi per la gestione del servizio:

- *node-red-start*, che avvia Node-RED e visualizza i log sulla console;
- *node-red-stop*, che arresta Node-RED;
- *node-red-restart*, che arresta e riavvia Node-RED;
- *node-red-log*, che visualizza i log del servizio.

È anche possibile fare in modo che il servizio venga avviato automaticamente al boot. Per farlo è sufficiente eseguire il seguente comando sulla console:

```
sudo systemctl enable nodered.service
```

Per disabilitare l'autostart al boot basta invece eseguire questo comando:

```
sudo systemctl disable nodered.service
```

NODE-RED EDITOR

Ora che Node-RED è correttamente installato sulla nostra Raspberri Pi, possiamo aprire l'editor; questo può essere aperto con un comune browser, puntando all'indirizzo nel quale il server sta girando. Quindi se vogliamo accedere "localmente", ossia dalla stessa Raspberry nella quale abbiamo installato Node-RED è sufficiente puntare alla porta 1880 del localhost: *http://localhost:1880*. Tuttavia nella maggior parte dei casi risulta più conveniente accedere a Node-RED da un altro PC, puntanto all'indirizzo IP della Raspberry nella nostra network locale sulla porta 1880, ossia *http://<hostname>:1880, dove <hostname>* rappresenta appunto l'indirizzo IP della Raspberry, che può essere recuperato dal nostro access point casalingo. Una volta aperto il browser all'indirizzo corretto e sulla porta 1880, dovremmo trovare una schermata simile a quella in Fig. 7. L'editor è composto essenzialmente dai seguenti quattro componenti:

- Header: contiene il pulsante di deploy, il menu principale, e, se la user authentication è abilitata, lo user menu.
- Workspace: è l'area di lavoro sulla quale i flows possono essere creati, inserendo nodi e connessioni tra di essi.
- Palette: contiene i nodi che possono essere utilizzati. Un nodo viene selezionato nella Palette e trascinato sul workspace per essere posizionato.
- Sidebar: contiene una serie di tools aggiuntivi che possono essere utilizzati dall'utente.

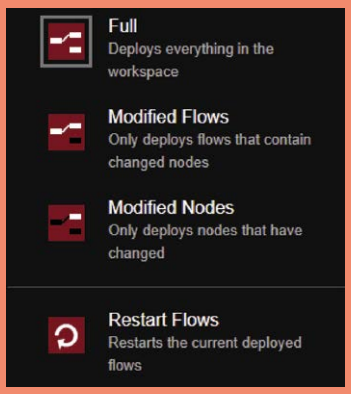


Fig. 8 - Differenti opzioni di deploy

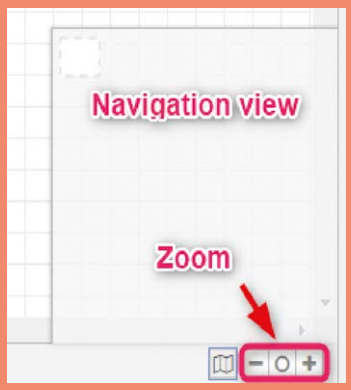


Fig. 9 - Footer del workspace con la Navigation view attiva.

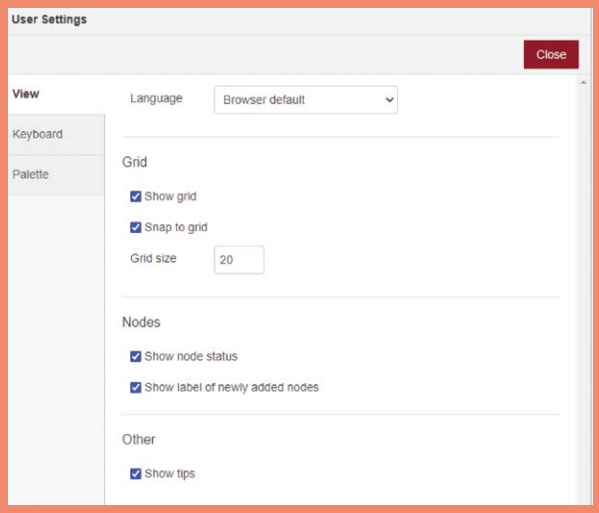


Fig. 10 - User Settings.

Analizziamo questi elementi più nel dettaglio per comprenderne meglio alcune carattersitiche:

Header

Come abbiamo visto, l'header contiene essenzialmente il menu principale, accessibile cliccando sull'icona con tre barre orizzontali ed il pulsante di deploy. Torneremo sul menu principale in seguito nelle puntate successive per illustrarne l'utilizzo. Per quanto riguarda invece il pulsante di deploy è importante sottolineare che la funzione di questo pulsante serve per rendere effettive le modifiche che abbiamo eseguito sul flow; sostanzialmente tutte le volte che modifichiamo un flow le informazioni vengono accumulate sul client ma non trasferite sul server che gira sulla Raspberry. Per far si che il flow lato server venga aggiornato deve appunto essere premuto il pulsante di deploy, che si colora di rosso se il flow ha delle modifiche di cui non è ancora stato fatto il deploy. Questo pulsante permette 3 differenti modalità di deploy (come evidenziato in Fig. 8):

- Full: viene eseguito il deploy di tutti flows sul workspace
- Modifed Flows: viene eseguito il deploy solo dei flow modificati
- Modified Nodes: viene eseguito il deploy solo dei nodi modificati

Workspace

Il workspace è la principale area di lavoro di Node-RED, dove i flows sono sviluppati prendendo nodi dalla Palette e collegandoli insieme. Il workspace ha una serie di tab in alto, corrispondenti ai vari flows che sono stati creati ed aperti.
Il footer del workspace ha alcuni tool di ausilio alla navigazione dei flows (Fig. 9). In particolare sono presenti pulsanti per eseguire zoom e de-zoom nel flow e per ripristinare il valore di zoom di default. Inoltre è disponile un pulsante per attivare la navigazione del flow, se questo dovesse essere talmente grande da non poter essere visualizzato per intero nella finestra.
È inoltre possibile seguire alcune customizzazioni del workspace tramite l'opzione "settings" del menu principale, come visibile in Fig. 10.

Palette

La Palette contiene la lista di tutti i nodi installati e disponibili all'uso. I nodi sono organizzati in categorie (ad esempio alcune delle categorie di base sono common, output e functions). Cliccando l'header di una categoria è possibile comprimere o espanderne la visualizzazione (inoltre due pulsanti presenti al fondo della palette, con il simbolo di una doppia freccia verso l'alto o verso il basso possono essere utilizzati per comprimere ed espandere tutte le categorie). Nella parte alta della palette è presente anche un input testuale che permette di filtrare la lista dei nodi per nome.

Sidebar

Infine la sidebar è una barra laterale che contiene una serie di tools aggiuntivi a disposizione dell'utente. I tool di base sono i seguenti:

- Information: permette di visualizzare informazioni sui nodi.
- Help: permette di visualizzare l'Help dei nodi.
- Debug: pemette di visualizzare i messaggi passati al nodo Debug.
- Configuration Nodes: permette di gestire i Configuration Nodes (un particolare tipo di nodo non visualizzato sul workspace).
- Context Data: permette di analizzare il contenuto dei context, un elemento che non è stato ancora presentato, e che verrà analizzato nella prossima puntata.

Node.JS

Node.js è un runtime system open source multipiattaforma orientato agli eventi per l'esecuzione di codice JavaScript, costruito sul motore JavaScript V8 di Google Chrome ed originariamente creato da Ryan Dahl nel 2009. Normalmente gli script JavaScript vengono utilizzati lato client, generalmente incorporati all'interno dell'HTML di una pagina web, venendo interpretati da un engine incorporato direttamente all'interno di un Browser. Node.js consente invece di utilizzare JavaScript anche per scrivere codice da eseguire lato server, ad esempio per la produzione del contenuto delle pagine web dinamiche prima che la pagina venga inviata al Browser dell'utente. Node.js in questo modo permette di implementare il cosiddetto paradigma "JavaScript everywhere" (JavaScript ovunque), unificando lo sviluppo di applicazioni Web intorno ad un unico linguaggio di programmazione (JavaScript).
Node.js ha un'architettura orientata agli eventi che rende possibile l'I/O asincrono. Questo design punta ad ottimizzare il throughput e la scalabilità nelle applicazioni web con molte operazioni di input/output, è inoltre ottimo per applicazioni web real-time (ad esempio programmi di comunicazione in tempo reale o browser game).
Node.js è un progetto di sviluppo open source distribuito gestito dalla Node.js Foundation e facilitato tramite il programma di progetti collaborativi della Linux Foundation.
Le aziende che supportano il programma includono GoDaddy, Groupon, IBM, LinkedIn, Microsoft, Netflix, PayPal, Rakuten, SAP, Voxer, Walmart e Yahoo!

Inoltre alcuni nodi contribuiscono alla sidebar aggiungendo pannelli specifici (è il caso della Node-RED Dashboard, che vedremo in una delle puntate successive).

USARE NODE-RED SU RASPBERRY PI

Ora che abbiamo introdotto i concetti base di Node-RED e abbiamo presentato l'editor, passiamo al primo semplice esempio pratico, in modo da evidenziare la semplicità che Node-RED consente di raggiungere nello sviluppo delle applicazioni. Ciò che vogliamo realizzare è il classico lampeggio di un LED, pilotandolo da un GPIO della Raspberry alla frequenza di 1Hz. Per realizzare questo semplice esempio ci bastano 3 nodi:

- un nodo Inject;
- un nodo Trigger;
- un nodo Rpi – Gpio Out.

Iniziamo posizionando il nodo Inject, prelevandolo dalla Palette all'interno della categoria common. Il nodo Inject (Fig. 11) è utilizzato per iniettare dati sul flow, di diverso tipo e con varie modalità. Noi lo utilizzeremo per generare un inject periodico che fornisca la periodicità di esecuzione al nostro flow.
Una volta posizionato il nodo va configurato, per dargli il comportamento desiderato. In questo caso noi vogliamo che il nodo Inject fornisca la periodicità. Per configurarlo dobbiamo cliccare due volte sul nodo con il tasto sinistro del mouse. Una volta eseguita questa operazione si aprirà

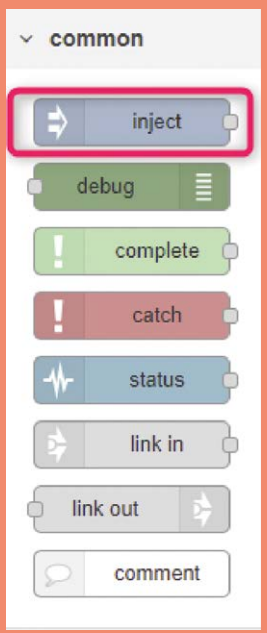


Fig. 11 - Nodo Inject.

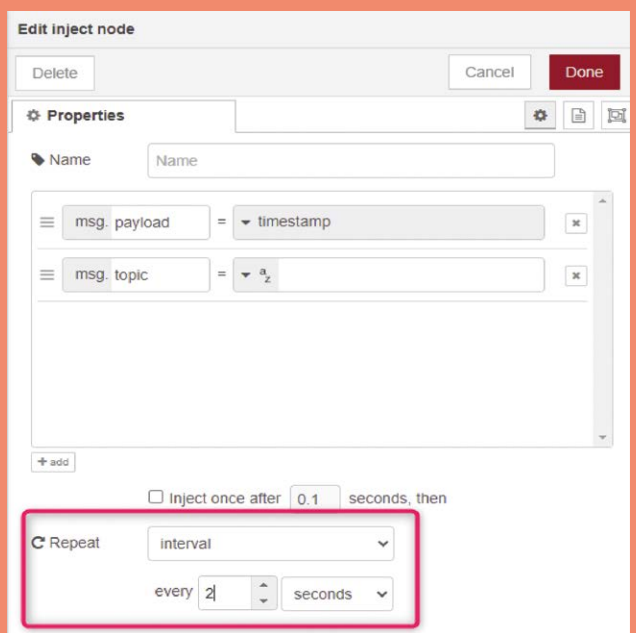


Fig. 12 - Configurazione del Nodo Inject.

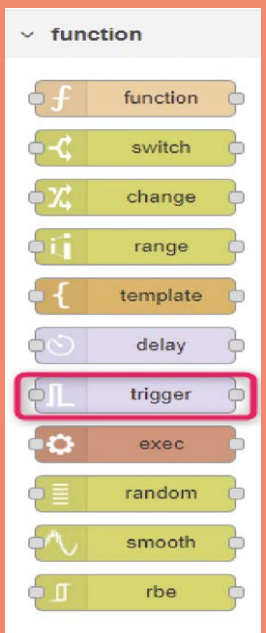


Fig. 13 - Nodo Trigger.

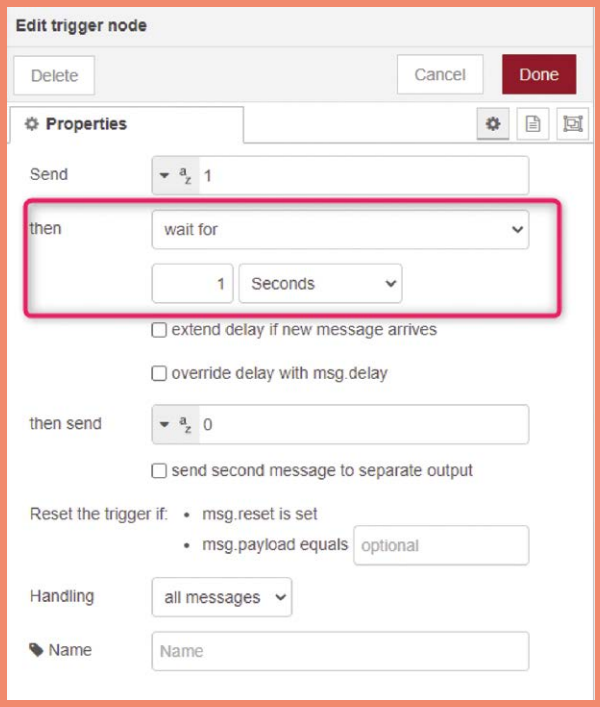


Fig. 14 - Configurazione del nodo Trigger

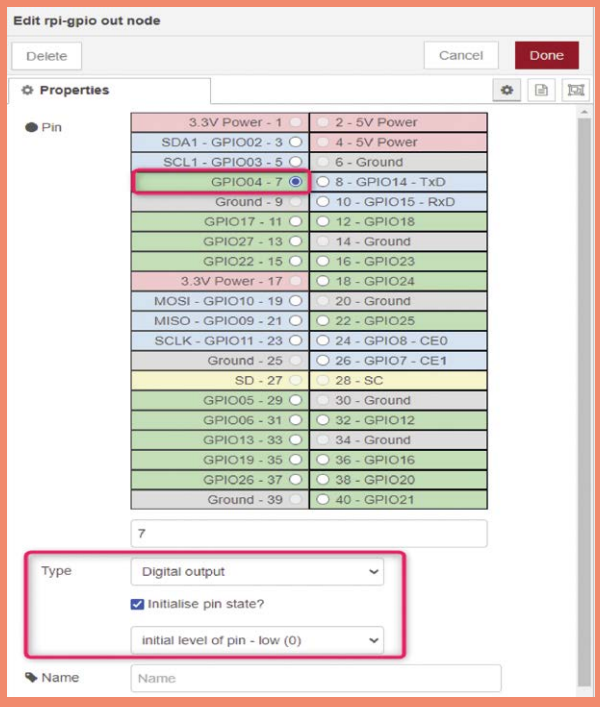


Fig. 16 - Configurazione del nodo rpi - gpio out

la finestra di Fig. 12, dalla quale è possibile configurare il nodo Inject. In questo particolare caso a noi basta configurare l'opzione Repeat, presente in basso, impostandola su Interval e indicando 2 secondi come intervallo.
Non ci interessa modificare il Payload del nodo, che in questo caso è un timestamp, perché non lo utilizzeremo per pilotare direttamente il LED, ma solo per fornire la periodicità di attivazione.
Una volta fatto clicchiamo su Done per chiudere. Ora possiamo inserire il nodo trigger, prelevandolo dalla categoria functions della Palette (Fig. 13).
La funzione del nodo trigger è quella di inviare un messaggio con un determinato payload quando attivato e, dopo un lasso di tempo configurato dall'utente, un secondo messaggio, sempre con un determinato payload. Questo è il nodo che noi utilizzeremo per generare l'informazione di controllo per il LED (prima on e dopo un secondo off). Apriamo la finestra di configurazione del nodo e modifichiamo il delay tra il primo e il secondo messaggio impostandolo ad 1 secondo (Fig. 14). In questo modo il nodo invierà, quando attivato, ossia alla ricezione di un messaggio in ingresso (il messaggio inviato dal nodo inject ogni 2 secondi), prima un messaggio contenente "1" e poi un secondo messaggio contenente "0". In questo modo abbiamo generato l'invio di un messaggio contenente "1" e dopo un secondo di un messaggio contenente "0" ogni 2 secondi, ossia abbiamo una serie di messaggi che rappresentano il "lampeggio" del nostro LED. Non ci rimane altro da fare che inserire un nodo che piloti direttamente il pin di GPIO cui vogliamo collegare il nostro LED.
Il nodo che fa al caso nostro è il nodo rpi – gpio out, presente nella categoria Raspberry Pi (Fig. 15).
Posizioniamo il nodo sul workspace ed apriamone la configurazione. Qui dobbiamo semplicemente selezionare il GPIO che vogliamo utilizzare (il GPIO 4 nell'esempio, ma può essere utilizzato qualsiasi GPIO adatto allo scopo) e il tipo di gestione (campo Type: Digital output) e, opzionalmente, se vogliamo inizializzare il pin ad un certo livello logico, come illustrato in Fig. 16.
A questo punto non ci rimane altro da fare che creare i collegamenti. La creazione dei collegamenti è molto semplice,

Fig. 15 - Nodo rpi - gpio out.

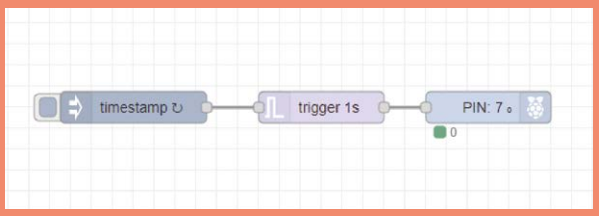


Fig. 17 - Flow per il lampeggio di un LED a 1Hz.

è sufficiente posizionarsi con il puntatore del mouse su una porta di un nodo cliccare e mantenere premuto il tasto sinistro, trascinare il puntatore sulla porta che si desidera collegare e rilasciare. Colleghiamo quindi l'output del nodo inject all'input del nodo trigger e l'output del nodo trigger all'input del nodo rpi – gpio out, come illustrato in Fig. 17. A questo punto possiamo premere il tasto deploy per mandare in esecuzione il nostro flow. Si noti che il nodo rpi-gpio out fornisce una indicazione dello stato fisico del pin (icona verde nell'angolo in basso a sinistra del nodo, con raffigurato il valore logico assunto dalla linea di I/O che stiamo utilizzando). In Fig. 18 è rappresentato lo schema delle connessioni da utilizzare, usando una breadboard, un LED e una resistenza da 1kΩ, più dei normali cavetti di collegamento da breadboard. Per i nostri esperimenti abbiamo utilizzato una Raspberry Pi 4 Model B, ma sostanzialmente qualsiasi modello di raspberry è utilizzabile (per maggiori dettagli sul nodo RPI gpio è possibile consultare la relativa pagina sul sito di Node-RED: *https://flows.nodered.org/node/node-red-node-pi-gpio*).

CONCLUSIONI

Siamo arrivati alla conclusione di questa prima puntata di questo corso Node-RED; in questa puntata abbiamo spiegato cos'è Node-RED ed il principio della programmazione flow-based. Inoltre abbiamo iniziato a prendere confidenza con l'editor e visto i primi semplici esempi su Raspberry PI. Nella prossima puntata inizieremo ad esaminare i più importanti nodi di base e vedremo alcune particolarità dell'ambiente, come ad esempio i context. □

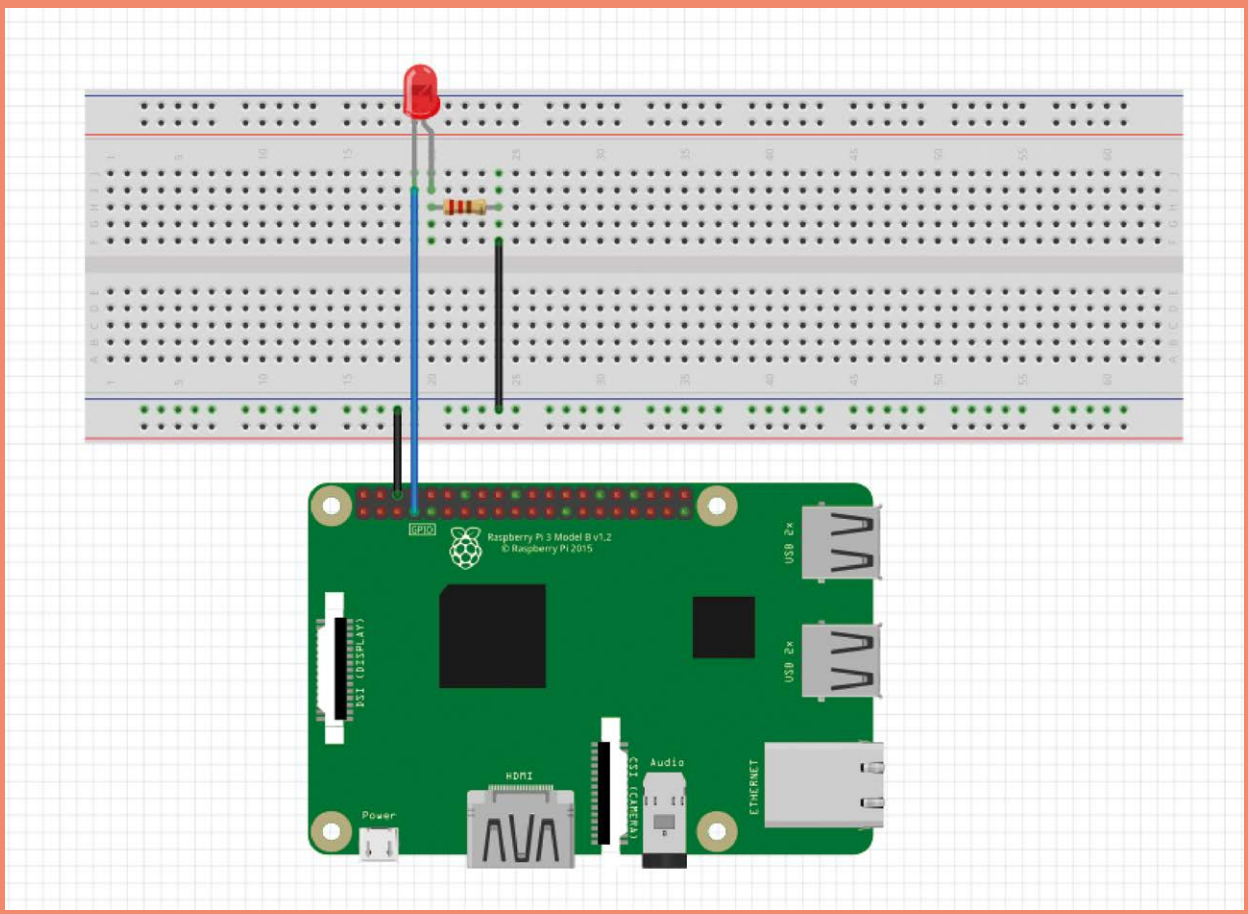


Fig. 18 - Schema dei collegamenti per il progetto di esempio.