

Session 1

Node-RED Basics to Design Event-Driven Applications



This work is licensed under a Creative Commons Attribution – NonCommercial – NoDerivatives 4.0 International License.

Objectives

- Install and discover the **Node-RED tool**

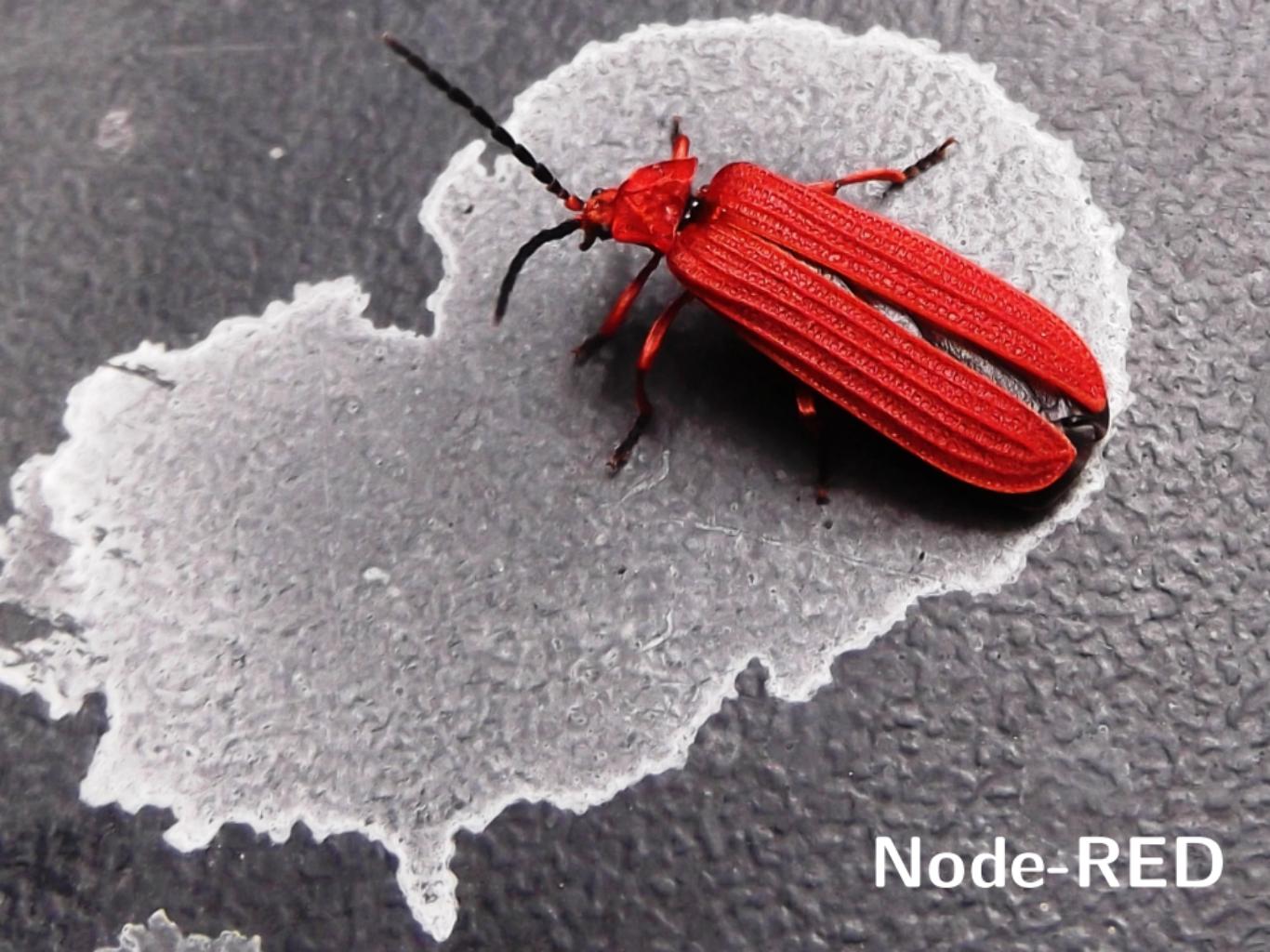
Write a first program with this visual programming tool

- Understand the **event-driven programming** model

And designing programs by defining flows with Node-RED

- Connect **hardware to the cloud** and make them interact

Analysing a simple use case with an Arduino

A close-up photograph of a bright red firefly beetle (Pyrochroa coccinea) resting on a piece of white, torn paper. The beetle is positioned diagonally, facing towards the top left. Its body is a vibrant, metallic red with a slightly textured surface. It has long, dark antennae and six visible legs. The background is a dark, textured gray.

Node-RED

Node-RED

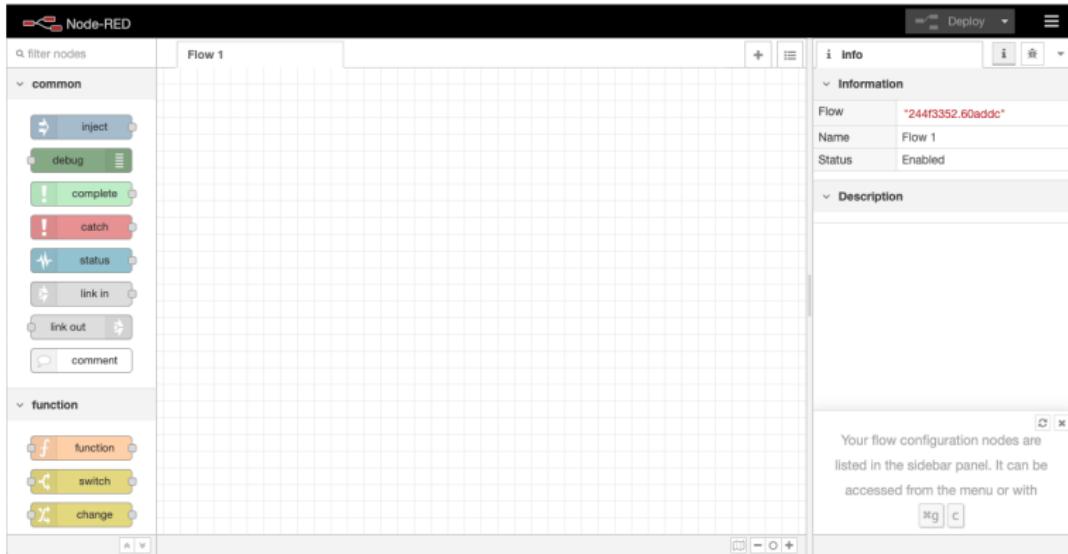
- Flow oriented and visual **programming tool** developed by IBM
 - Visual editor directly integrated in the web browser
 - Hardware can be easily wired with APIs and cloud services
- Tool based on the **Node.js** JavaScript runtime
 - JavaScript functions are created graphically or with code
 - Flows are stored in a single JSON file



Node-RED Platform

- Node-RED platform directly available from a **browser**

Launched on localhost on port 1880, by default



Hello World (1)

- First **Hello World** program with Node-RED
 - Generating a string when clicking on a button
 - Writing the content of the string on the debug sidebar
- Dragging **two nodes** onto the workspace from the palette
 - One inject node as source and one debug node as destination
 - Linked to make the data flow from one node to the other



inject Node

- **Inject a message** with a given type in a flow
 - Message type can be string, number, boolean, JSON, etc.
 - Manual injection or injection at regular interval

Payload

Topic

Inject once after seconds, then

Repeat

Name

Note: "interval between times" and "at a specific time" will use cron.

"Interval" should be 596 hours or less.

See info box for details.

debug Node

- **Display a message** in the debug sidebar
 - Can display the payload or the complete message object
 - Output can be sent to the debug sidebar or system console

The screenshot shows a configuration interface for the Node.js 'debug' module. At the top left is a 'Output' button with a list icon. To its right is a dropdown menu set to 'msg. payload'. Below this is a 'To' section with two options: 'debug window' (which is checked) and 'system console' (which is unchecked). Further down is an optional section for 'node status (32 characters)' which is also unchecked. At the bottom is a 'Name' section with a heart icon and a text input field containing the word 'Name'.

Output	msg. payload
To	<input checked="" type="checkbox"/> debug window <input type="checkbox"/> system console <input type="checkbox"/> node status (32 characters)
Name	Name

Hello World (2)

- Deploy the program to build and launch it

Then click on the left of the inject node to inject a message



Flow



Event-Driven Paradigm

- Node-RED follows the **event-driven** programming paradigm
 - Capture, communication, processing and persistence of event
 - Opposed to the traditional request-driven model
- An **event** represents any substantial change in state

Can be a change in hardware as well as in software
- An event gives rise to an **event notification** message

Sent by the system to notify another part of the event occurrence

Event-Driven Architecture

- Event-driven applications can be created in many languages
This is an approach and not a programming language
- Several advantages of an **event-driven architecture**
 - Less coupling and easier maintenance
 - Better scalability and evolutivity
 - Good choice for modern distributed systems
- **Loose coupling** because producers do not know consumers

And event does not know consequences of its occurrence

EDA Application

- Ability to design **strong, flexible and adaptable** architecture

The event producer does not need to know the event consumer

- No **deterministic response time** is guaranteed

Way easier to create realtime responsive architectures

- Typically used for **IoT applications** with data collection

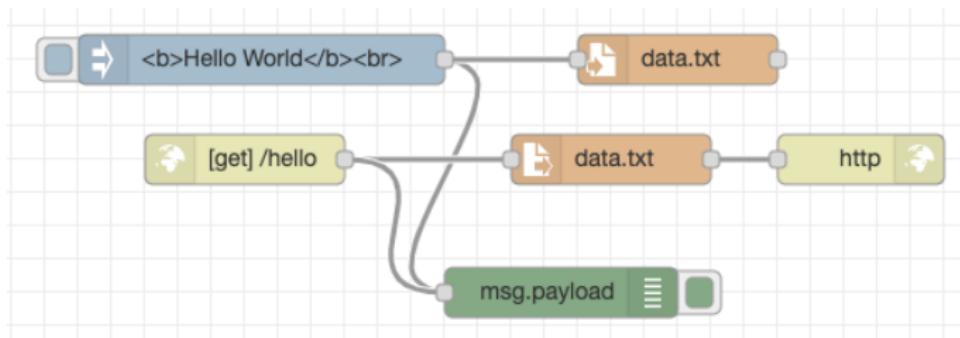
Collected data are ingested by a software platform in the cloud

Event-Driven Architecture Model

- The **Pub/Sub model** is based on subscriptions
 - Messaging infrastructure with subscriptions on event streams
 - A published event is sent to all the subscribers
- The **event streaming model** is based on an event log
 - All the events are written to a single event log
 - The log can be read at any time, starting from any point

HTTP Server (1)

- Creating an HTTP end-point to propose a **web service**
 - http in node to serve a content on a URL
 - file node to read the content of a file
 - http response node to send the HTTP response to the client



file Node

- Write the payload of a message to a text file
 - Can append, overwrite or delete a file
 - Can configure options to create directory, set encoding, etc.

Filename	<input type="text" value="data.txt"/>
Action	<input type="button" value="append to file"/>
<input checked="" type="checkbox"/> Add newline (\n) to each payload?	
<input type="checkbox"/> Create directory if it doesn't exist?	

Encoding	<input type="text" value="default"/>
-----------------	--------------------------------------

Name	<input type="text" value="Name"/>
-------------	-----------------------------------

Tip: The filename should be an absolute path, otherwise it will be relative to the working directory of the Node-RED process.

http in Node

- Set up an HTTP end-point used to create a web service
 - Define the desired URL and HTTP method
 - HTTP server listening on same URL and port as Node-RED

Method	GET
URL	/hello
Name	Name

file in Node

- **Read the content** of a text file
 - Choose the path to the file to be read
 - The file can be read at once or generate several messages

Filename	<input type="text" value="data.txt"/>
Output	a single utf8 string
Encoding	default
Name	Name

Tip: The filename should be an absolute path, otherwise it will be relative to the working directory of the Node-RED process.

HTTP Server (2)

- Filling the file and then **connecting to the HTTP server**

Data read from the file is displayed on the browser

The screenshot shows two panels. On the left, a terminal window displays the following log entries:

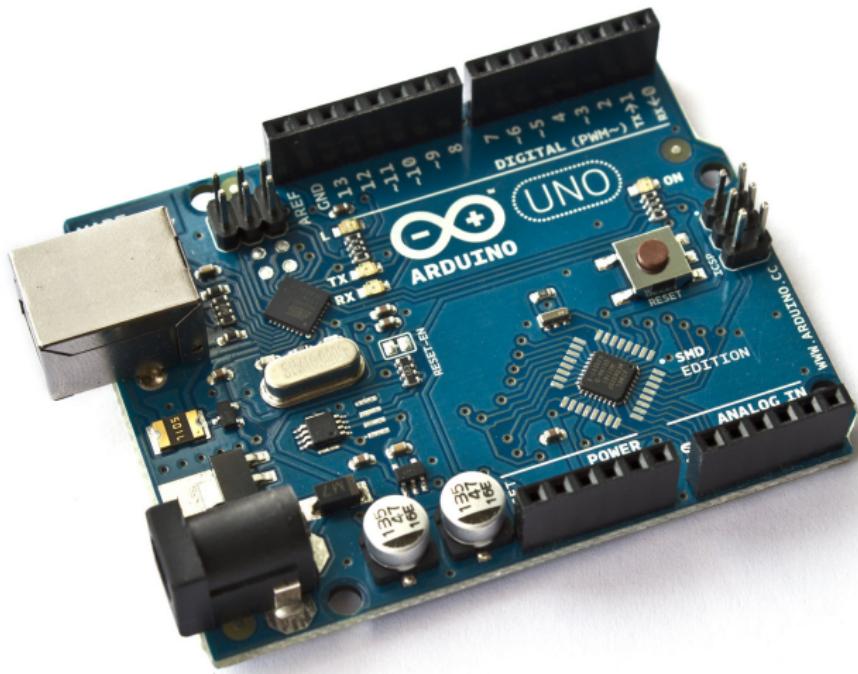
```
02/02/2020 à 22:11:53 node: 8e5ba2db.65e9e
msg.payload : string[22]
"<b>Hello World</b><br>"

02/02/2020 à 22:11:53 node: 8e5ba2db.65e9e
msg.payload : string[22]
"<b>Hello World</b><br>"

02/02/2020 à 22:11:57 node: 8e5ba2db.65e9e
msg.payload : Object
{ empty }
```

On the right, a browser window shows the URL `127.0.0.1:1880/hello`. The page content is:

Hello World
Hello World

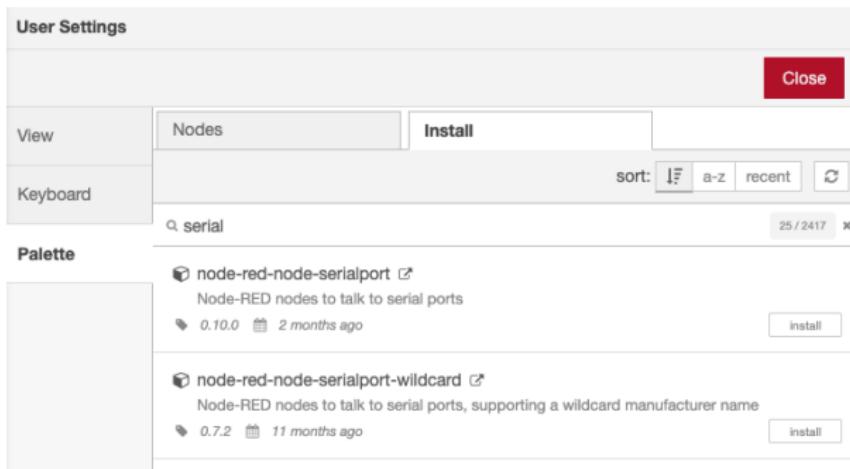


Connecting Hardware

Installing Module

- Installing a module through the **Manage Palette** menu

Adding the node-red-node-serialport module, for example



Arduino Hello World (1)

- Arduino program writing Hello World on the **serial port**

Watch carefully for the path to the serial device and baud rate



```
sketch_jul01a §
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Hello World");
  delay(1000);
}
```



serial in Node

- Read data from a local serial port

Configuring carefully the parameters of the serial connection

Serial Port /dev/tty.usbmodem14201

Settings

Baud Rate ▼ 9600	Data Bits 8	Parity None	Stop Bits 1
DTR auto	RTS auto	CTS auto	DSR auto

Input

Optionally wait for a start character of , then

Split input on the character \n

and deliver ascii strings

Output

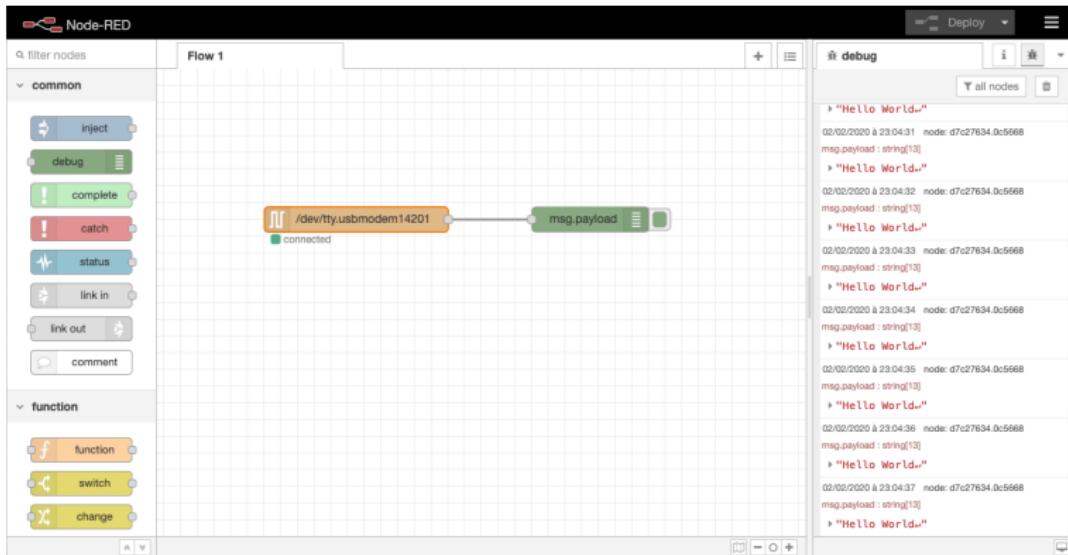
Add character to output messages

Request

Default response timeout 10000 ms

Arduino Hello World (2)

- The serial in node is **properly connected** to the serial port
All incoming messages are directly sent to the debug sidebar



References

- Node-RED. *Tutorials: Creating your first flow*, retrieved on February 2, 2020.
<https://nodered.org/docs/tutorials/first-flow>
- Red Hat (2019). *What is event-driven architecture?*, September 27, 2019.
<https://www.redhat.com/en/topics/integration/what-is-event-driven-architecture>
- Telmo Subira Rodriguez (2018). *Understanding Event-Driven Architectures (EDA): the paradigm of the future. What will EDA contribute to your digital business?*, September 9, 2018.
<https://medium.com/drill/understanding-event-driven-architectures-eda-the-paradigm-of-the-future-7ae632f056bb>
- Caleb Keller (2017). *A Fun Introduction to Node-RED*, September 11, 2017.
<https://blog.spg.ai/a-fun-introduction-to-node-red-4d14d8bdbcb6b>

Credits

- J Brew, June 2, 2013, <https://www.flickr.com/photos/brewbooks/8950778490>.
- SkHazard, February 2, 2018, <https://en.wikipedia.org/wiki/File:Node-red-icon.png>.
- Colin, June 26, 2016, <https://www.flickr.com/photos/colinsd40/27303249684>.
- Justus Blümer, April 4, 2012, <https://www.flickr.com/photos/justusbluemer/7050677125>.