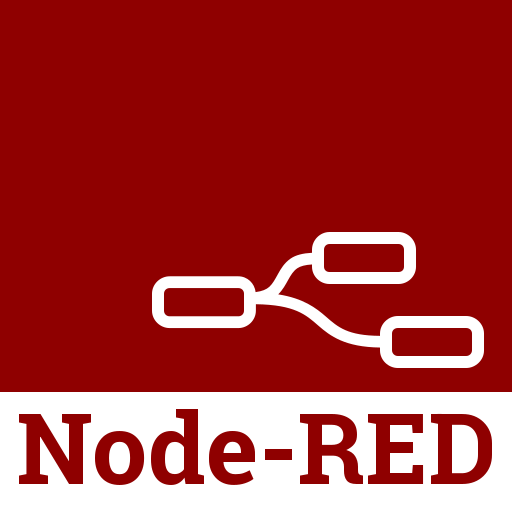
Node-RED



Semester 3 Communication

Node-RED Workshop

Oct 19, 2022.

Student: Andre Sanao

Course: Technology

Table of Contents

[Introduction 3](#_Toc117197413)

[Research 3](#_Toc117197414)

[Testing 3](#_Toc117197415)

[Task 1 3](#_Toc117197416)

[Task 2 4](#_Toc117197417)

[Task 3 6](#_Toc117197418)

[Conclusion 8](#_Toc117197419)

[Bibliography 9](#_Toc117197420)

# Introduction

In this lecture, we learned how to work with Node-RED. Node-RED is used for developing fast prototypes in IoT. It is also used for connecting hardware’s, API’s and services for example web and cloud. Node-RED is based on Node.js and is programmed using visual flows. In Embedded Systems, it is used to capture data and send it to the GUI on the web.

# Research

In this section will be all information that was given in the lecture. Node-RED works with visual blocks. These blocks are represented as JSON texts and communicate by sending messages. The basic blocks contain ID and a payload in each message. Most of them have only an input, output or both that can be connected with wires. By double clicking on the block, you can see all the information you can configure.

Some useful feature about Node-RED is that plugins can be downloaded to connect with hardwares such as MQTT or a GUI Dashboard.

# Testing

In the workshop, we needed to work around Node-RED by downloading Node.js first before Node-RED. To connect to Node-RED, you need to enter “localhost:1880” in the URL.

## Task 1

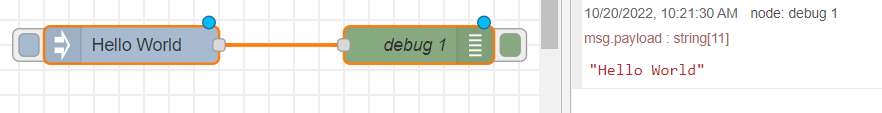
We needed to send the message “Hello World” to the payload and use the debugger to see the message. The debug output needs to be enabled to use the debugger. After enabling it, you can press “Deploy” to upload the message. (Figure 1)

Figure Block code for Hello World

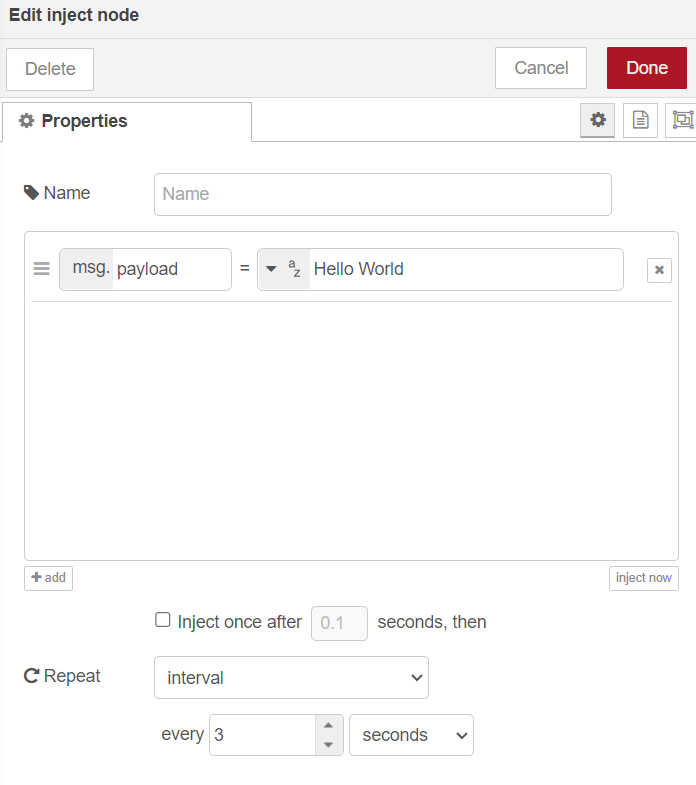
After that we had to make it automatically print every 3 seconds into the debugger. This can be configured by double clicking the inject block. (Figure 2)

Figure Config for interval counter

## Task 2

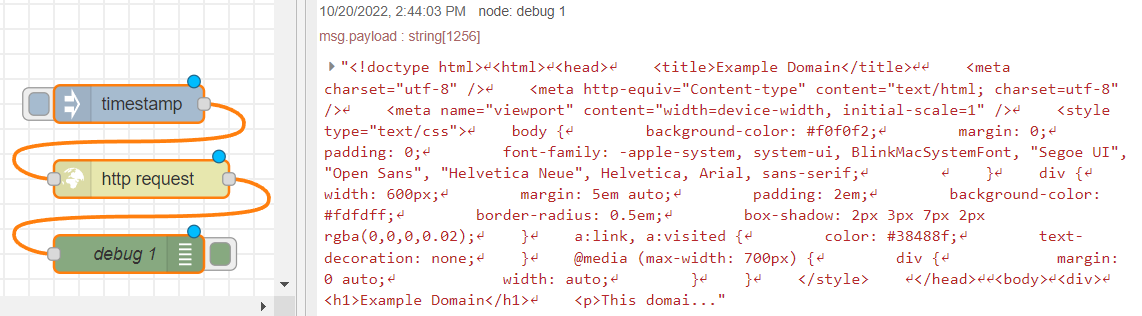
In this task, we are to explore on our own with the use of the “http” blocks. The first part is to output the contents of a webpage. I used “example.com” as the URL and outputted its webpage contents using the “http” block. (Figure 3)

Figure http request block for URL

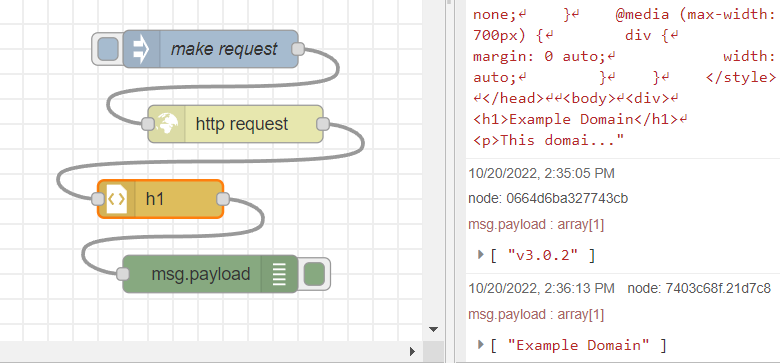
The second part is to output only the html <title> of the URL using the “html” block. The title is filtered out using the “html” block. (Figure 4)

Figure html block for title filter

For the third part, the “function” block is used to print an even number of timestamps that is generated by the inject block. (Figure 5)

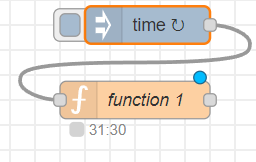


Figure function block for timestamp

The code uses the JavaScript language to use the Date(). To get the even numbers, modulo was used to print only numbers that are even. (Figure 6)

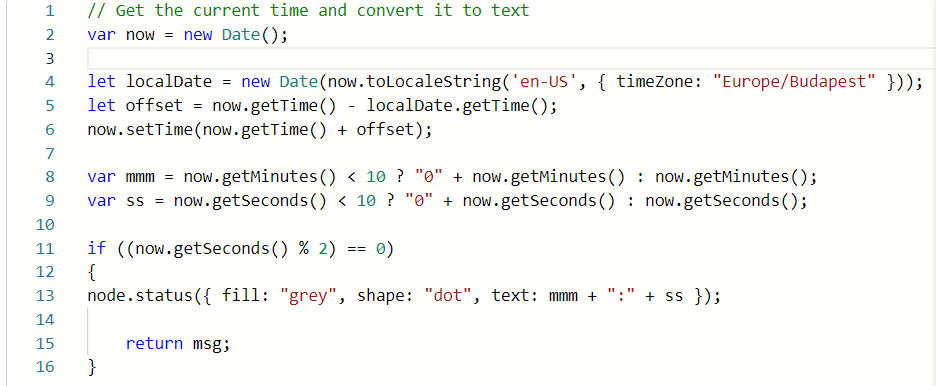


Figure JavaScript code for timestamp

The last part of this task is to print “hello world”. But when this string enters the function block, all the characters turn into uppercase. This requires us to use JavaScript again to code the characters. The function “.toUpperCase()” is used to make the msg payload into uppercase. (Figure 7) (Figure 8)



Figure JavaScript code for uppercase

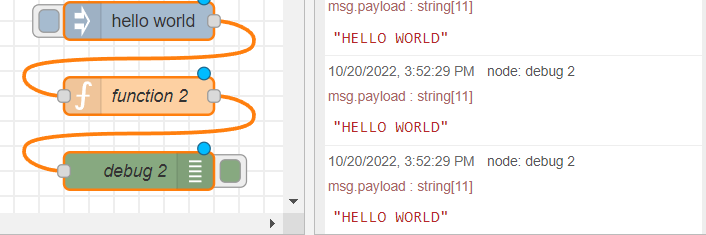


Figure Output for the uppercase function

## Task 3

In this task, we are going to explore the MQTT module. MQTT is used to connect the hardware to the internet with for example an ESP32. Mobile or laptop devices subscribes to the MQTT broker to get data. The broker publishes the data and sends it to the subscribers. There can be multiple hardware’s connected and to get this specific hardware, you need to enter the right topic.

To subscribe to a broker, you need to use the “MQTT in “ block. In this task we communicate with “test.mosquitto.org” with the port 1880. This URL has unsecured connection and should not be used to enter sensitive data because someone else might be subscribed and steal these data.

The inject block is used to send a message payload to the broker. The topic that is used in this task is “nr\_workshop/greetings”. (Figure 9)

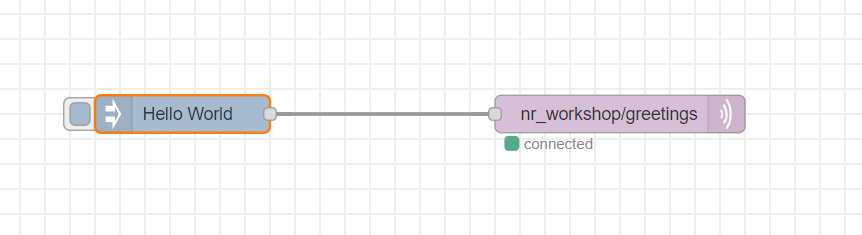


Figure block flow for MQTT

With this block flow, we can inject messages to the topic. The broker will then publish the message to the Node-Red Dashboard. (Figure 10)



Figure greetings from the inject block

For the last part, we are to use an ESP32 and publish the hall sensor to the Node-Red Dashboard. We used the assignment code that is provided and read the hall sensor values. These values are published to the topic “nr\_workshop/hallSensor/<your\_name>”. The Workshop Dashboard is subscribed to the “nr\_workshop/hallSensor/#” topic, so we should be able to see the output on the dashboard. (Figure 11)

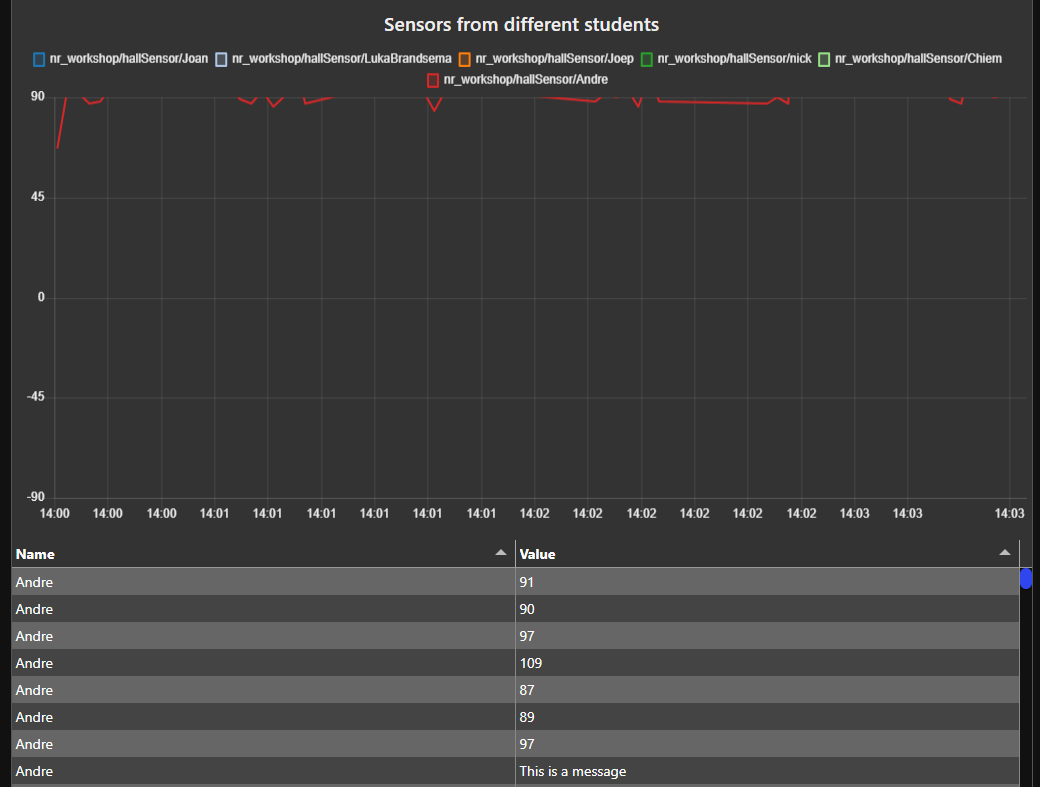


Figure Hall sensor values in dashboard

# Conclusion

In conclusion, Node-Red is very useful for connecting hardware and monitoring it online. It does require knowledge of JavaScript to code. The Dashboard can be used for future project to output data values of the sensors and it is mandatory for some upcoming projects.

# Bibliography

*Running node-red locally* (no date) *Node*. Available at: https://nodered.org/docs/gettingstarted/local#installing-with-npm (Accessed: October 20, 2022).

Steve *et al.* (2022) *Node-red HTTP request node for Beginners*, *Steves Node-Red Guide -*. Available at: https://stevesnoderedguide.com/node-red-http-request-node-beginners (Accessed: October 20, 2022).

*Simple get request* (no date) *Node*. Available at: https://cookbook.nodered.org/http/simple-get-request (Accessed: October 20, 2022).

*Javascript string touppercase()* (no date) *JavaScript String toUpperCase() Method*. Available at: https://www.w3schools.com/jsref/jsref\_touppercase.asp (Accessed: October 20, 2022).