

SPRINT 3-REPORT

Creating Node Red Service

Node and online services in new and interesting ways. -RED is a programming tool for wiring together hardware devices, APIs

Go to your Node-RED flow editor

This instance is running as an IBM Cloud application, giving it access to the wide range of services available on the platform. [Learn how to customise Node-RED](#)
More information about Node-RED, including documentation, can be found at nodered.org.

Creating a Flow

Node-RED

Flow 3

filter nodes

common

- inject
- debug
- complete
- catch
- status
- link in
- link call
- link out
- comment

function

- function
- switch
- change

info

Search flows

Flows

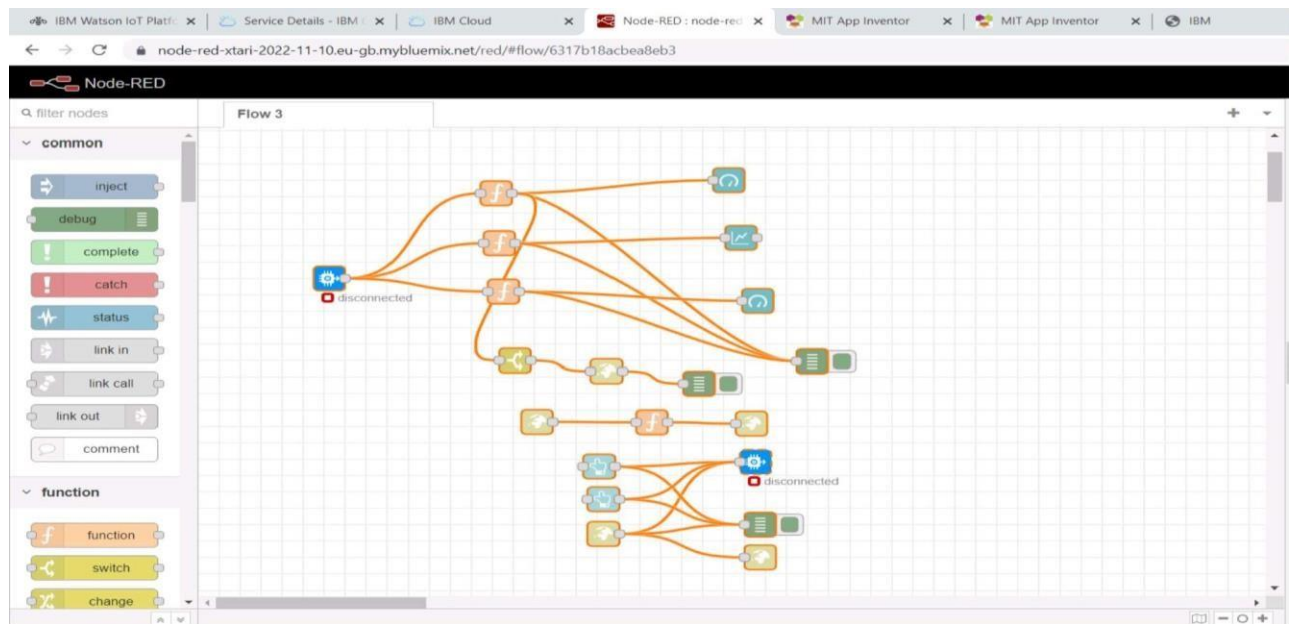
- Flow 3
- Subflows
- Global Configuration Nodes

Flow 3

Flow "6317b18acbea8eb3"

Pressing enter will edit the first node in the current selection

Creating the Project Flow



Configuring the Node with the Credentials

API Key

latest

Device Event

Device Event

Authentication

O

API Key

All or 12345

Input Type

event type e.g. status

Device Type

All or o All or json

Device Id

0

Event

IBM 10T

Format

IBM 10T

@ oos Name

registered

Service

Enabled

Cancel

Done

Flow 3

Info

Search flows

Flows

Flow 3

Subflows

Global Configuration Nodes

IBM 10T

Node

"35c78269c8e2e20?"

ibmiot in

Hold down ctrl when you click on a node to add or remove it from the current selection

https://node-red-xtan-2022-11-

*editor-tab-properties

SPRINT 2 Smart f...docx

SPRINT 2 Smart fa...pdf

Milestone&activity

Milestone&activity...pdf

Milestone&activi...docxMilestone&activi.docx

Show all x

Connected to the Watson Platform

The screenshot shows the Node-RED web interface in a browser. The top bar includes tabs for IBM Watson IoT Platform, Service Details - IBM, IBM Cloud, Node-RED - node-red, MIT App Inventor, and IBM. The main workspace displays a flow diagram with several nodes connected. On the left, the 'common' and 'function' node palettes are visible. On the right, the 'debug' console shows a log of messages. The messages include timestamps, node IDs, and payloads, such as:

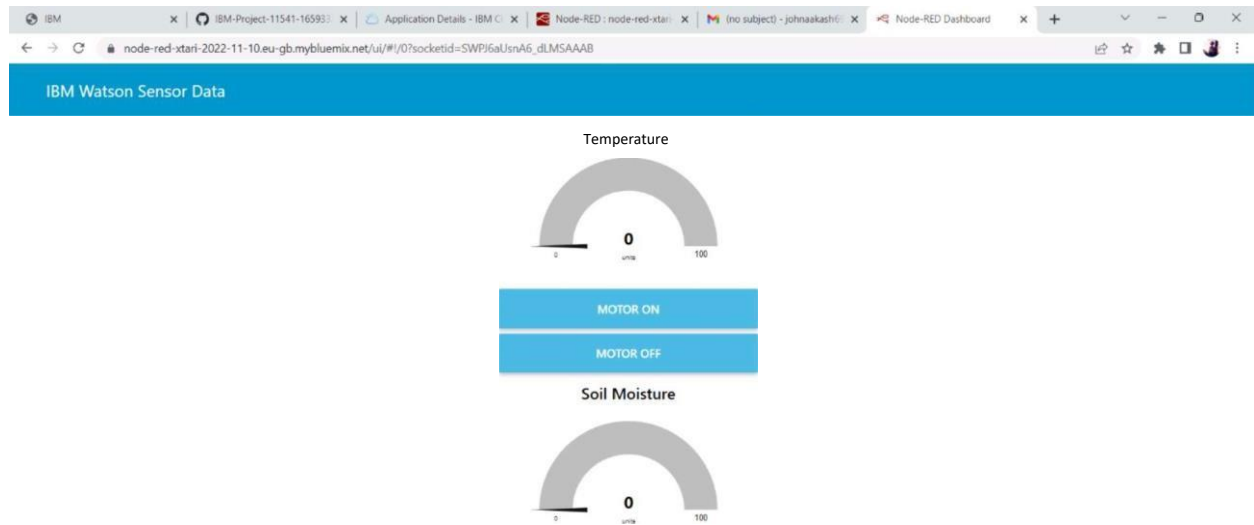
```
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoron" }  
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoroff" }  
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoron" }
```

Receiving the data from the Watson Platform and MIT App Inventor

This screenshot is a zoomed-in view of the debug console from the previous image. It shows a sequence of messages received from the Watson Platform and MIT App Inventor. The messages are timestamped and include node IDs and payloads. The payloads are objects containing a 'command' property, such as 'motoron' and 'motoroff'. The messages are as follows:

```
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoroff" }  
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoron" }  
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoroff" }  
11/11/2022, 1:51:11 AM node: IBM IoT  
msg: string[51]  
"[ApplicationClient:publish] Client is not connected"  
11/11/2022, 1:51:11 AM node: 0e1f5b2ab4b3941  
msg payload: Object  
{ command: "motoron" }
```

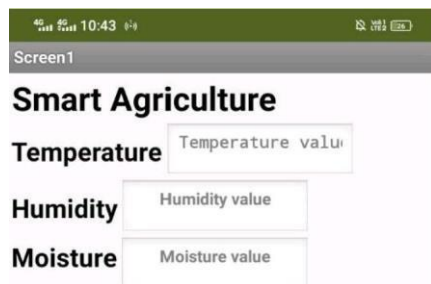
Web Application UI (for reference)



MIT App as Input and Node Red as Output

MOTOR ON	MOTOR OFF
-------------	--------------





User Interface Displaying the Random Data

