Plat'Home
TECHNOLOGY to serve you.

OpenBlocks IoT Family
Node-RED Starter Guide

Ver.3.3.0

Plat'Home Co., Ltd.

■ About trademarks

- Linux is a trademark or registered trademark of Linus Torvalds in the United States and/or other countries.
- Company and product names mentioned in this Set-up Guide may be trademarks or registered trademarks of their respective companies.
- Product names and other proper nouns in this Set-up Guide are trademarks or registered trademarks of their respective companies.

■ Before using this product

- No reproduction of this material is allowed without written permission of Plat'Home Co., Ltd.
- Content and information contained within this material may be changed or updated without prior notice.
- We consistently aim to keep the content in this material as precise as possible. However, should any errors in descriptions, etc. be noticed, please contact Plat'Home Co., Ltd. The latest version of this material can be downloaded from our website.
- While using this product, please be aware that it is not designed or assumed for use in fields where there is a risk to life.
- Regardless of the aforementioned, in no event will Plat'Home be liable for any special, incidental, indirect or consequential damage arising out of use of this product, including but not limited to damage to profits or loss.

目次

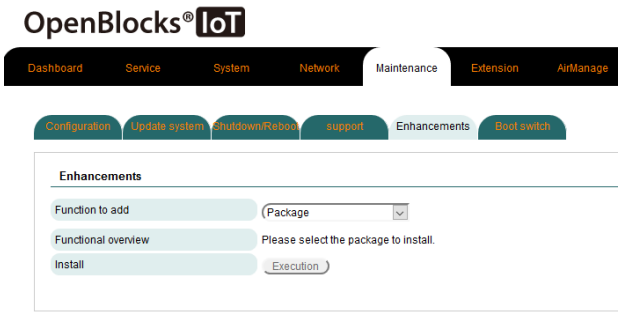
Chapter 1 General

This manual explains how to use Node-RED that can be installed in OpenBlocks IoT Family. The installed Node - RED is prepared as a candidate for the destination by the IoT data control function, and it is supposed to correspond to the implementation of edge computing and the cloud which is not compatible.

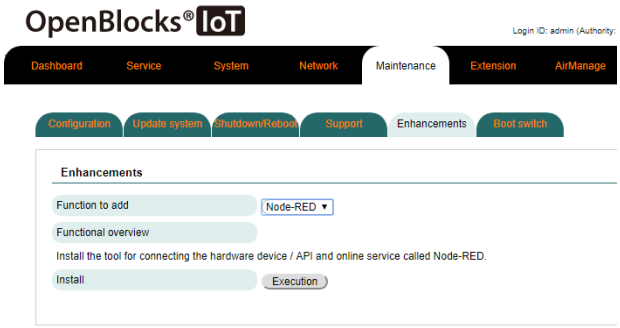
Chapter 2 Advance preparation

2-1. Installation of Node-RED

At the time of shipment from our factory, Node-RED is not installed in this product. To install Samba, using the **[Maintenance]-[Enhancements]** tab.



When choosing the **[Maintenance]** tab of WEB UI and clicking on the **[Enhancements]** tab, it is possible to choose a package for extensions.



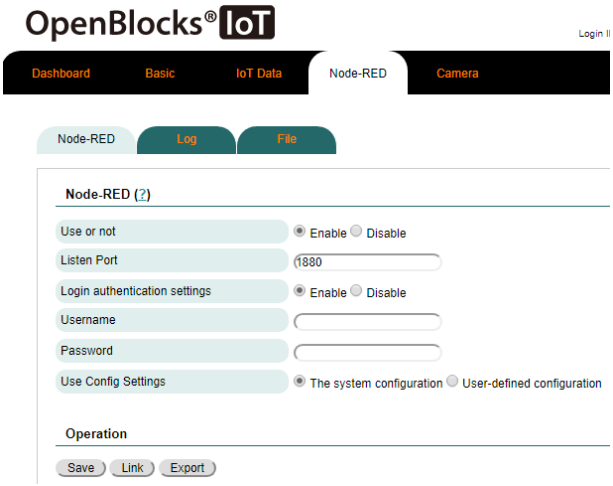
From the pull-down menu showing a list of packages to be installed, choose "Node-RED". Press the Execution button to install the program.

After completing installation, the unit will require rebooting to make the installation effective. Choose the Shutdown/Reboot tab from the Maintenance tab to reboot the unit.

2-2. Node-RED startup settings

When the Node-RED package has been installed, link of Node-RED function will be displayed in the **[Service]-[Basic]** tab.

Please press this Node-RED link. As a result, we will transition to the **[Node-RED]** tab.



It can configure the operation of Node-RED using the **Node-RED** menu in the **[Node-RED]- [Node-RED]** tab.

<u>Node-RED</u>

Use or not :

To use Node-RED, choose "Enable." If not, choose "Disable."

Listen Port :

Specify the port number for accessing Node-RED. Normally, It do not need to change it from the default of 1880.

Login authentication setting :

To use login authentication, select "Enable" and specify the username and password.

Use Config Setting :

Configure the operation of Node-RED, It can switch the configuration file to this system or user edited one.

When "User defined config" is selected, tab for configuration editing is displayed.

Node-RED will start and stop by click the save button.

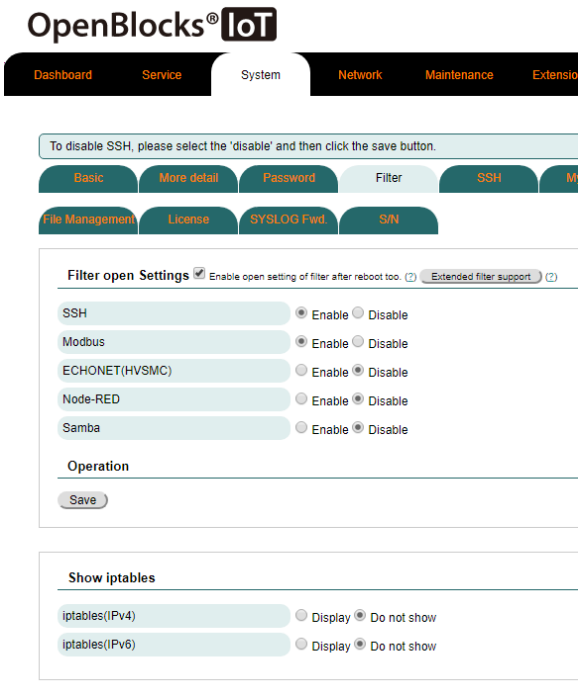
*By click the link button you can access the active Node-RED.

However, please use the URI proxy function when accessing via AirManage.

2-3. Packet filtering for Node-RED

In order to In order to access the Node-RED, it is necessary to open the communication port used by Node-RED.

In the **Filter open Settings** menu in the **[System]-[Filter]** tab, set Node-RED to "Enable."



By default the filter is applied so that it can not be accessed for Samba.

Please set it to "Enable" and press save button.

For security reasons, please close the filter after setting up Node - RED.

2-4. In coming packet filters

In coming packet filters of OpenBlocks IoT Family are not open except for ports required for system operation, such as access to Web UI and time synchronization.

When using a node waiting for a remote connection such as a TCP input node and receiving it from the outside, it is necessary to separate the packet filter separately as necessary.

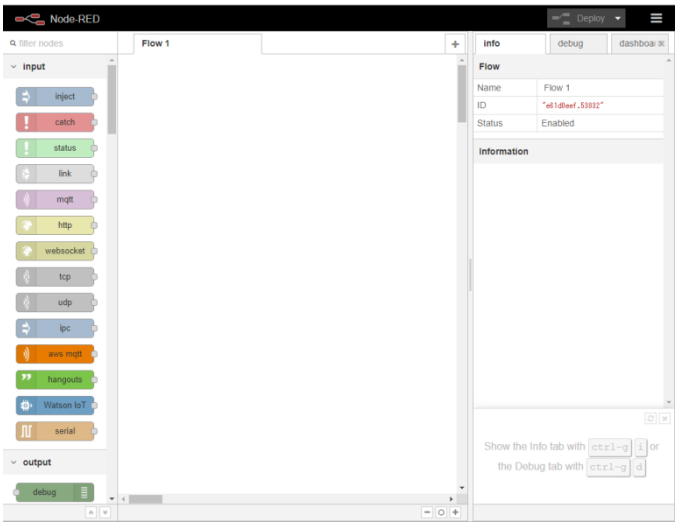
For operation of packet filter, please edit extended filter configuration file using **Edit extended filter configuration file** menu in the [System]-[Fileter] tab.

Chapter 3 Brief description of Node-RED

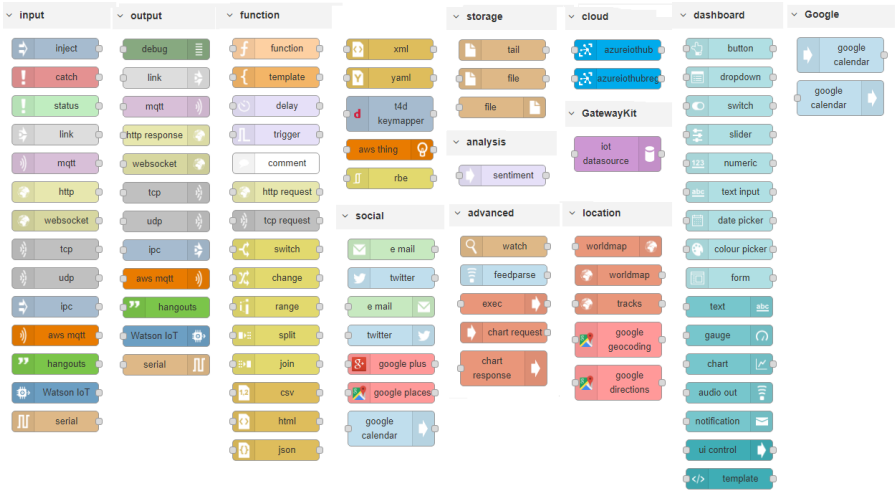
Access to dashboard of Node-RED is done by browser using 1880 port as default.
Therefore, the URL for accessing dashboard of Node-RED via the default Wi - Fi is as follows.

http://192.168.254.254:1880/

When accessing dashboard of Node-RED, the following screen will be displayed in the initial state. (When login authentication setting is not done)

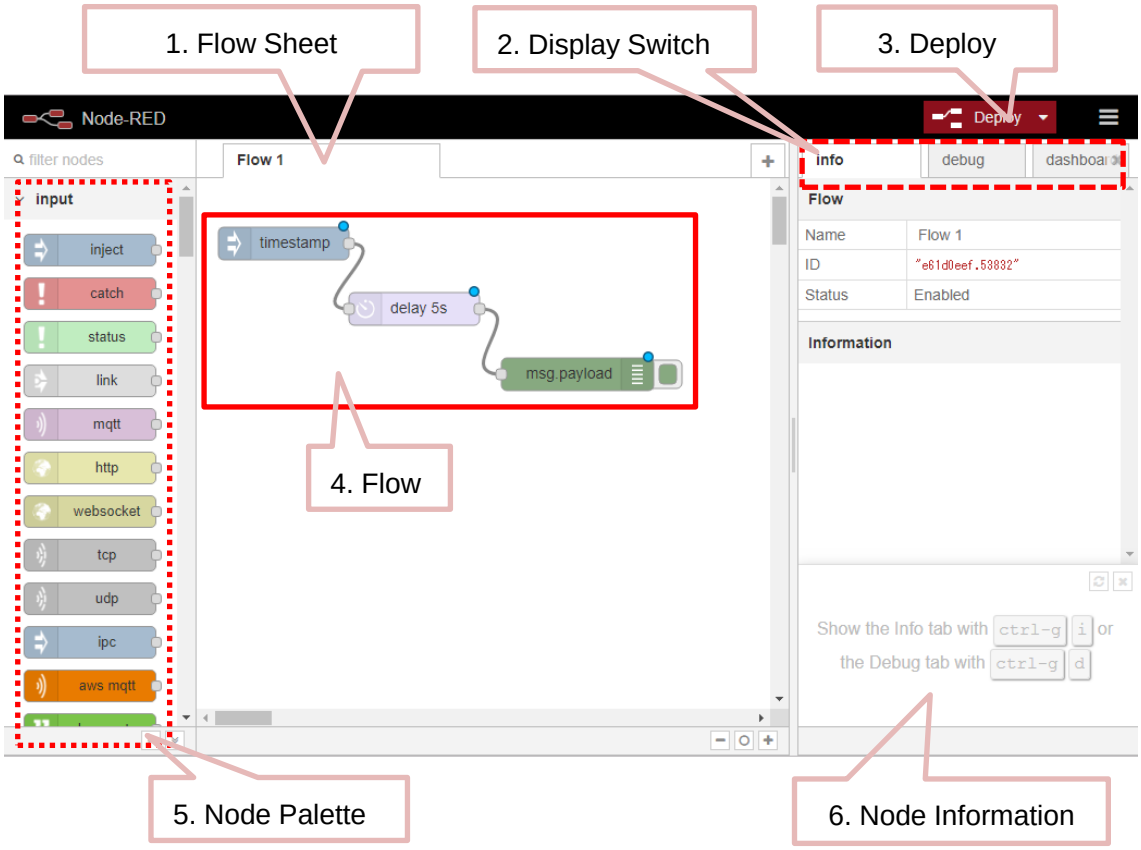


The nodes of input, output, processing etc. prepared by default for this product are as follows.



3-1. Screen composition of Node-RED

The screen composition of Node - RED as follows.

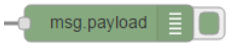


#	Item	Description
1	Flow Sheet	A workspace describing the processing flow.

2	Display Switch	Switch display of node information/debug information.
3	Deploy	Click the Deploy button to deploy the process flow described on the sheet.
4	Flow	Define the flow of data (process flow) by placing and connecting nodes.
5	Node Palette	Palette of nodes used for configuring the processing flow.
6	Node Information	Displays node information or debug information.

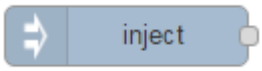
3-2. Types of Node

In Node - RED, there are roughly divided nodes with the following connector arrangement.

Right connector		**Both connector**		Left connector
				
Event activation When the start condition is satisfied, the data is transmitted to the right hand node.	➡	**Data processing** Receive data from the left hand and process, then send the processed data to the right hand node.	➡	**Event end** Receives data from the left hand, performs the defined processing and ends.

As described above, data is processed from left hand to right hand.

3-3. Input Node

	By click the button on the left side of the node, timestamp etc set to the node will be input data or event.
	An error occurred in a node on the same sheet is regarded take as input data or event.
	The status of the node on the same sheet take as input data or event.

link	The output of any link output node take as input data or event.
mqtt	Subscribe to the MQTT broker, wait for the publish data, and make it an input data or event.
http	Waits for an HTTP request and make it an input data or event.
websocket	Waits for connection by Websocket and make it an input data or event.
tcp	Waits for connection by TCP socket and make it an input data or event.
udp	Waits for connection by UDP socket and make it an input data or event.
ipc	Waits for connection by UNIX domain socket and make it an input data or event.
aws mqtt	Subscribe to the MQTT broker of AWS IoT, wait for the publish data, and make it an input data or event.
hangouts	Listen for data from Google hangouts and make it an input data or events.
Watson IoT	Subscribe to the MQTT broker of Watson IoT, wait for the publish data (device command), and make it an input data or event.
serial	Waits for input from the serial interface and make it an input data or event.

3-4. Output Node

debug	Displays the input data as debug information.
link	Outputs input data to one input link node.
mqtt	Publish input data to MQTT broker.
http response	Outputs the input data as a response to the input to the HTTP input node.
websocket	Output the input data to the Websocket server.

tcp	Output the input data to the TCP server.
udp	Output the input data to the UDP server.
ipc	Output the input data to the UNIX domain socket.
aws mqtt	Publish input data to MQTT broker of AWS IoT
hangouts	Output the input data to Google hangouts.
Watson IoT	Publish input data to MQTT broker of Watson IoT.
serial	Output the input data to the serial interface.

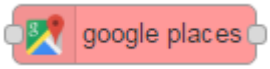
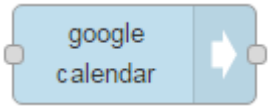
3-5. Function Node

function	Process input data with JavaScript and outputs it.
template	It formats the input data and outputs it.
delay	Outputs the input data after a specified time.
trigger	Specify timeout for input data and output two data.
comment	Add a comment to the flow.
http request	An HTTP request is made to the specified URL for the input data and the response is output.
tcp request	It makes a TCP connection to the specified server for the input data and outputs the response.
switch	Outputs input data to different nodes according to the specified branch condition.
change	It sets/changes/deletes or moves attributes of input data and outputs it.

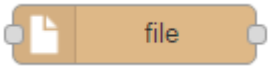
range	Change the scale of input data and outputs it.
split	Divides the input data by the specified character and outputs it.
join	Join input data and outputs it.
csv	Convert CSV format data and JavaScript Object to each other.
html	Convert HTML formatted data and JavaScript Objects to each other.
json	Convert JSON format data and JavaScript Object to each other.
xml	Convert data in XML format and JavaScript Object.
yaml	Convert data in YAML format and JavaScript Object.
t4d keymapper	Convert input data to input data for Toami for DOCOMO.
aws thing	Specify Thing Shadow control of AWS IoT.
rbe	It outputs only when the input data changes.

3-6. Social Node

e mail	Waits for an E-mail, and make it an input data or event.
twitter	Waiting for messages from Twitter, and make it an input data or event.
e mail	Send the input data to the specified e-mail address.
twitter	Outputs input data to Twitter.
google plus	Interact with the Google+ API to get information about people, activities, and comments.

	Utilizes the Google Places API in order to find and learn more about local establishments.
	Return the next event in a Google Calendar.

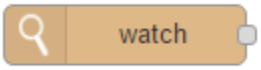
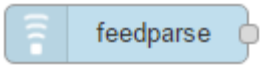
3-7. Storage Node

	Takes the end of the specified file as input data.
	Open the file which specified in the input data and output its contents.
	Output data to the specified file.

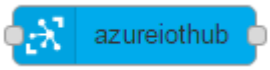
3-8. Analysis Node

	Empathy analysis (positive/negative/neutral) of input data using AFINN-111 word list and outputs it.

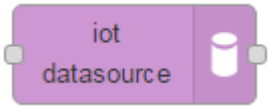
3-9. Advanced Node

	It monitors directory or file updates and make it an input event.
	Monitored RSS/Atom and detect updates of Web contents and, make it an input event.
	Executes the command of the specified system and returns its output.
	Request a graph drawing on Google chart.
	Output graph chart of Google chart.

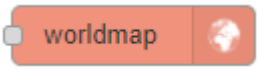
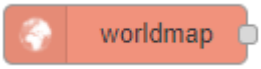
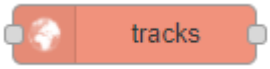
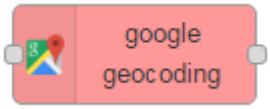
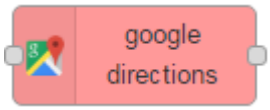
3-10. Cloud Node

	Outputs the input data to Azure IoT Hub.
	Register the device which specified in the input data to the Azure IoT Hub.

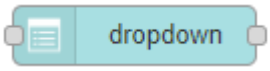
3-11. GatewayKit Node

	Output the input data to the dashboard application.

3-12. Location Node

	Plots "things" on a web map. Needs an internet connection.
	Receives events from a worldmap web page.
	Creates tracks lines based on a specified number of previous locations.
	Utilizes the Google Geocoding API to allow conversion of addresses to coordinate sets, and vice versa.
	Utilizes the Google Directions API to provide directions between the supplied origin and destination.

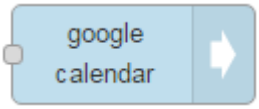
3-13. Dashboard Node

	Adds a button to the user interface.
	Adds a dropdown select box to the user interface.

switch	Adds a switch to the user interface.
slider	Adds a slider widget to the user interface.
numeric	Adds a numeric input widget to the user interface.
text input	Adds a text input field to the user interface. Mode can be regular text, email or color picker.
date picker	Adds a date picker widget to the user interface.
colour picker	Adds a colour picker to the dashboard.
form	Adds a form to user interface.
text	Will display a non-editable text field on the user interface.
gauge	Adds a gauge type widget to the user interface.
chart	Plots the input values on a chart. This can either be a time based line chart, a bar chart (vertical or horizontal), or a pie chart.
audio out	Plays audio or text to speech (TTS) in the dashboard.
notification	Shows msg.payload as a popup notification or OK / Cancel dialog message on the user interface.
ui control	Allows dynamic control of the Dashboard.
template	The template widget can contain any valid html and Angular/Angular-Material directives.

3-14. Google Node

google calendar	Waits the events of Google Calendar (schedule notification) and make it an input data or event.

	Create an entry in a Google Calendar.

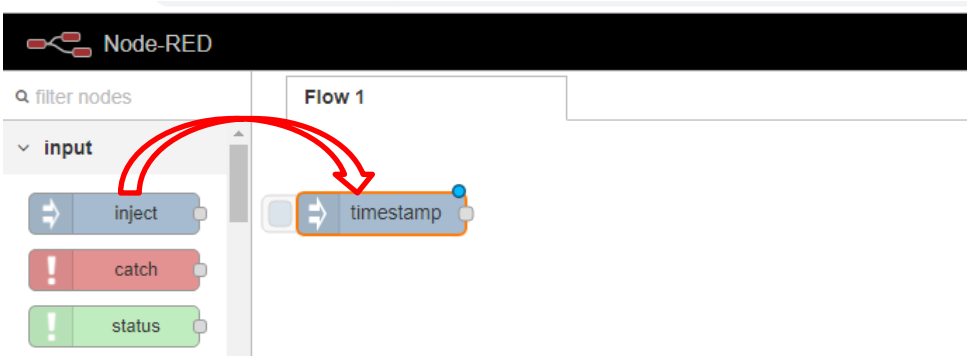
Chapter 4 Node operation example

4-1. Visualize location information

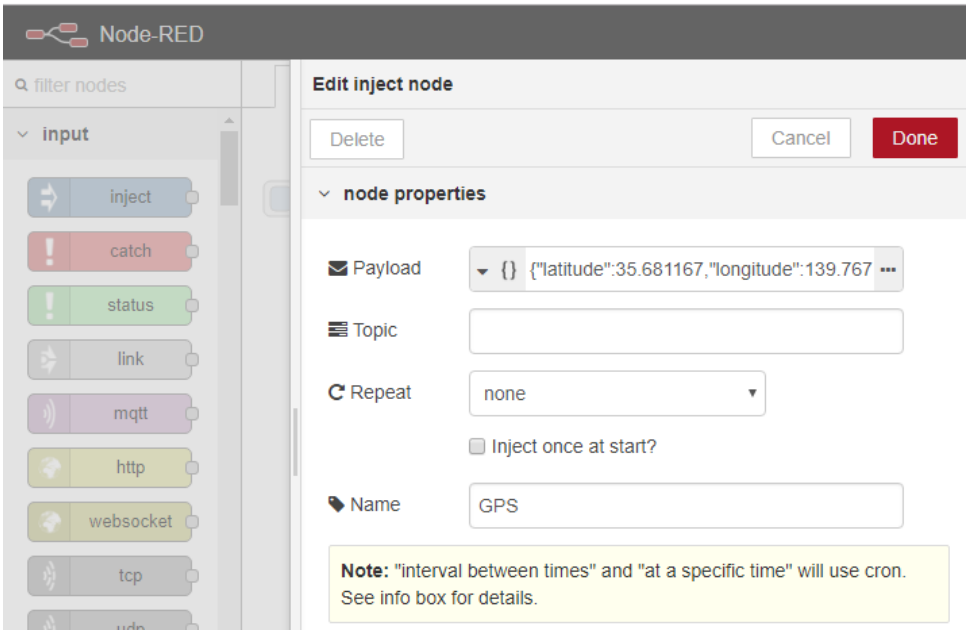
1. Access dashboard of Node-RED. (See Chapter 3)

   http://192.168.254.254:1880/

2. Drag the Inject node from the Input Nodes palette and drop it onto the sheet.



3. Double-click the placed Inject node and edit the properties of the node.



3a. Enter “GPS” in the Name field.

3b. Select “JSON” as the type of payload with ▼pull down menu in Payload field.

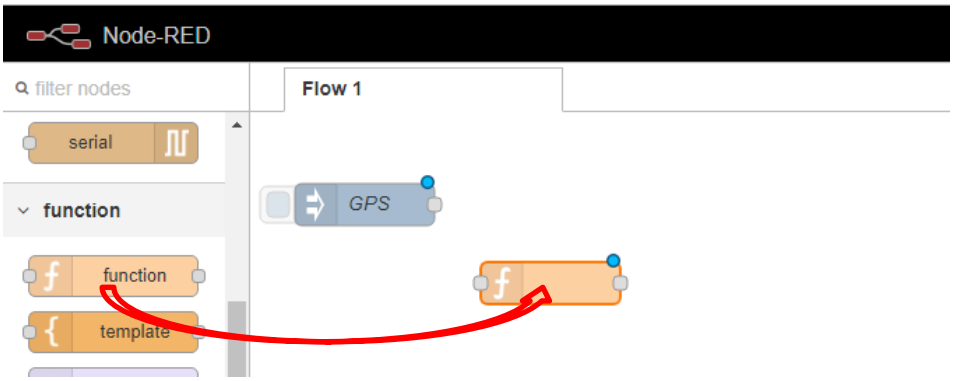
3c. Click the “...” button in Payload field then write the following to the displayed window

and click the “Done” button.

```
{
        "latitude": 35.681167,
        "longitude": 139.767052
}
```

3d. Click the ”Done” button.

4. Drag the function node from the Function Nodes palette and drop it onto the sheet.



5. Double-click the placed function node and edit the properties of the node.

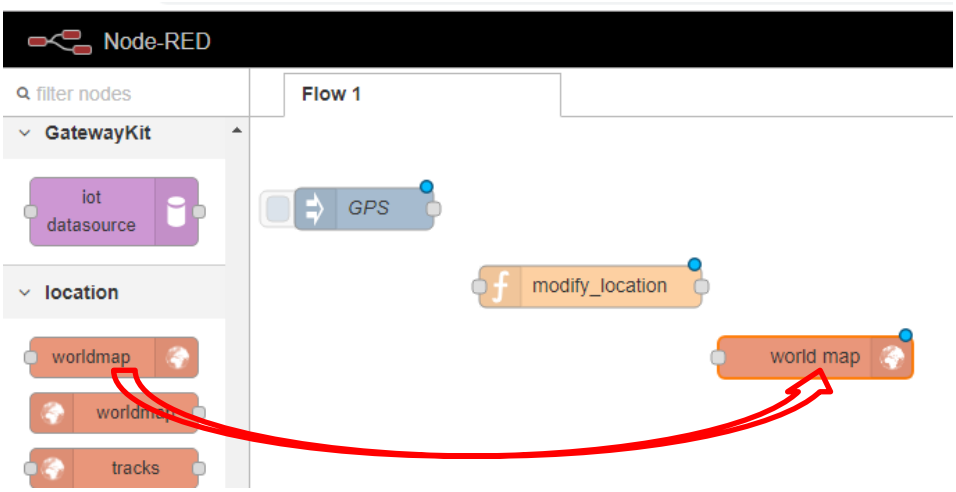
5a. Enter “modify_location” in the Name field.

5b. Write the following to the Function filed.

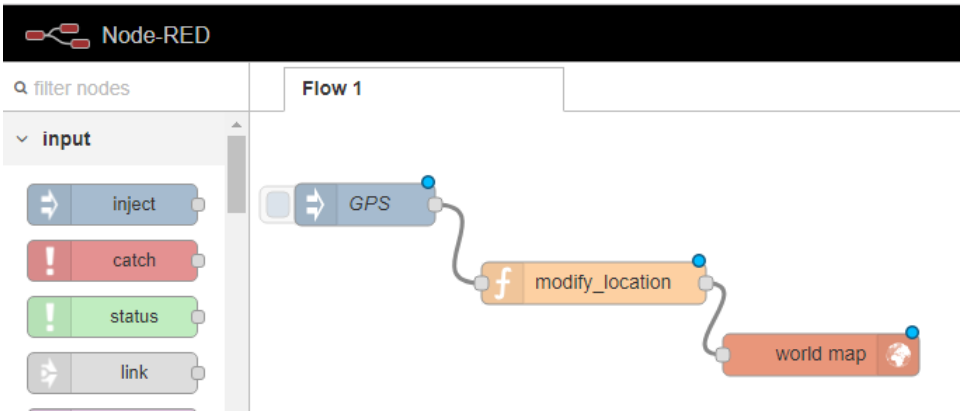
```
//var res = JSON.parse(msg.payload);
var res = msg.payload;

msg.payload = {
    name: "Tokyo sta." ,
    lat:  res.latitude,
    lon:  res.longitude
} ;

return msg;
```

5c. Click the ”Done” button.

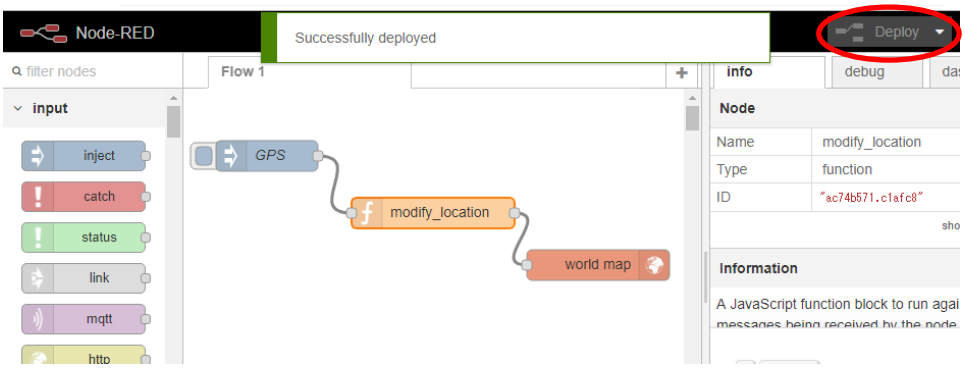
6. Drag the worldmap node from the Nodes palette and drop it onto the sheet.



7. Connect between Inject node and function node, function node and worldmap node.



8. Click the ”Deploy” button.

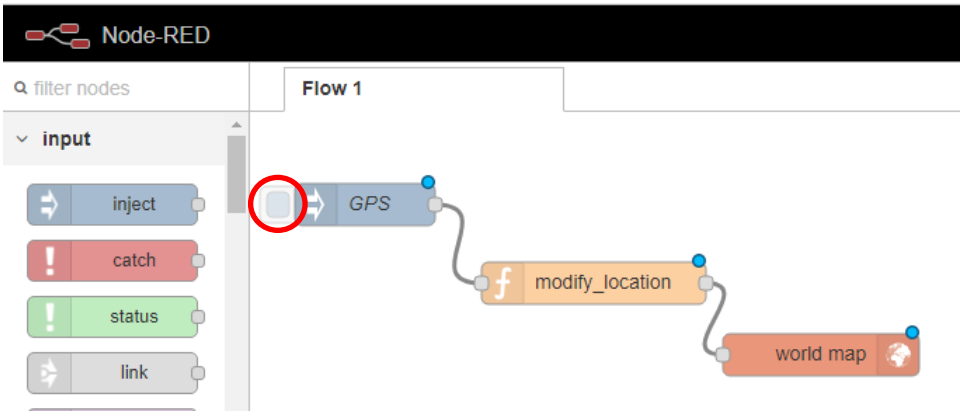


When Deploy completes, the background of the Deploy button changes from red to black

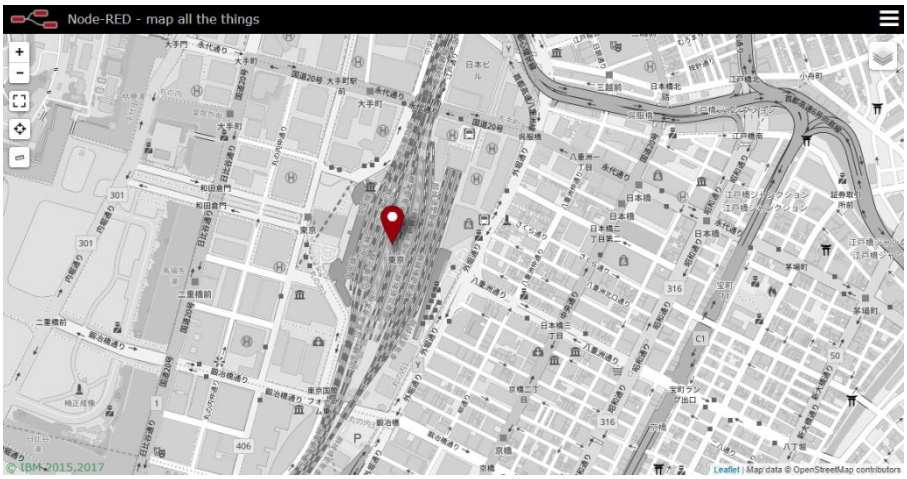
9. Access the following URL on another tab of the browser.

   http://192.168.254.254:1880/worldmap/

10. Click the left button of the Inject node.



11. It will be plotted at the Tokyo station in Japan on the map displayed on another tab.

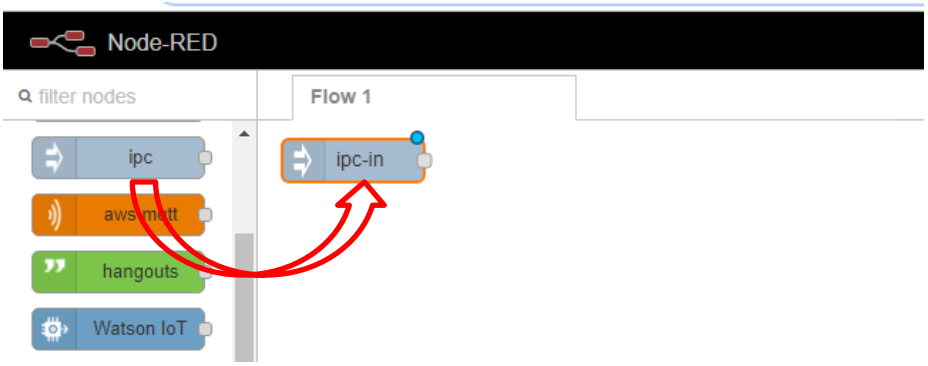


4-2. Giving global positioning information to beacon data

This section gives an example to add global positioning information of GPS acquired using the LTE module (for NTT docomo and KDDI) or BWA module to beacon data collected by the IoT data control function.

*KDDI and NTT docomo are domestic carrier in Japan

1. Drag the ipc node from the Input Nodes palette and drop it onto the sheet.

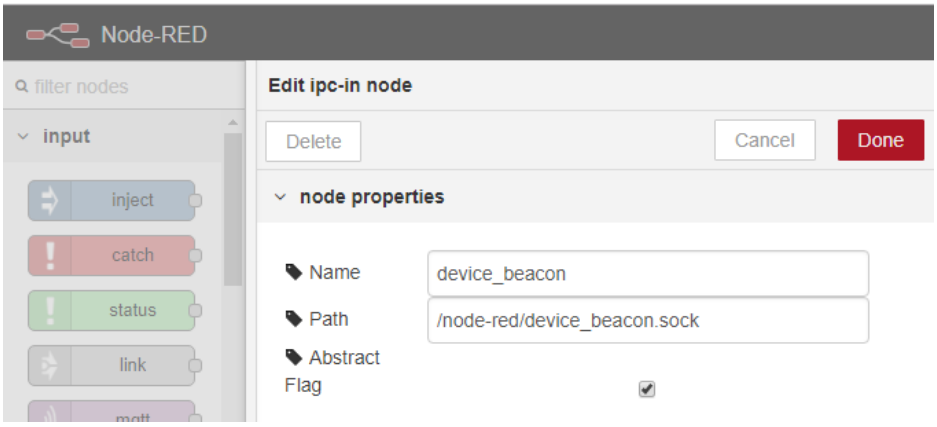


2. Double-click the placed ipd node and edit the properties of the node.

   2a. Enter “device_beacon” in the Name field.

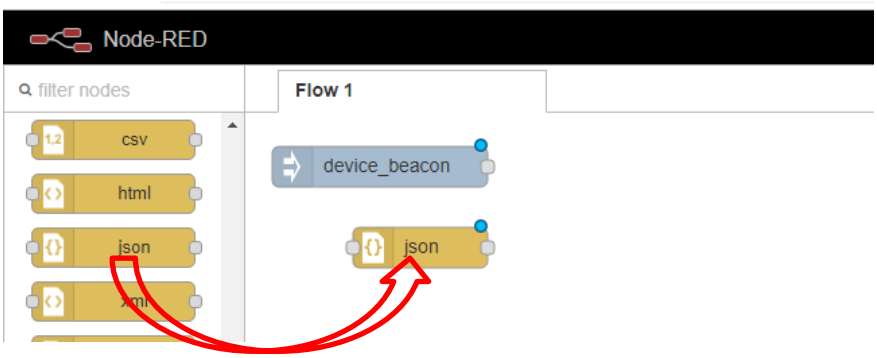
   2b. Enter “/node-red/device_beacon.sock” in Path filed.

   2c. Check Abstract Flag check box.

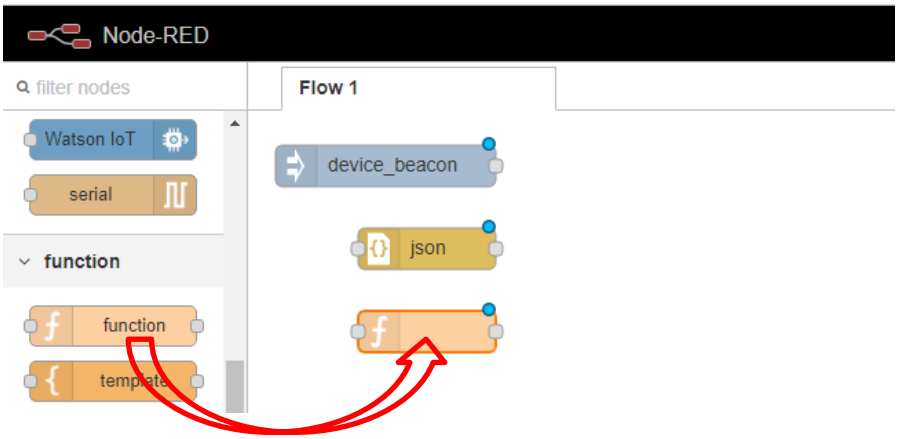


   2d. Click the ”Done” button.

3. Drag the json node from the Function Nodes palette and drop it onto the sheet.

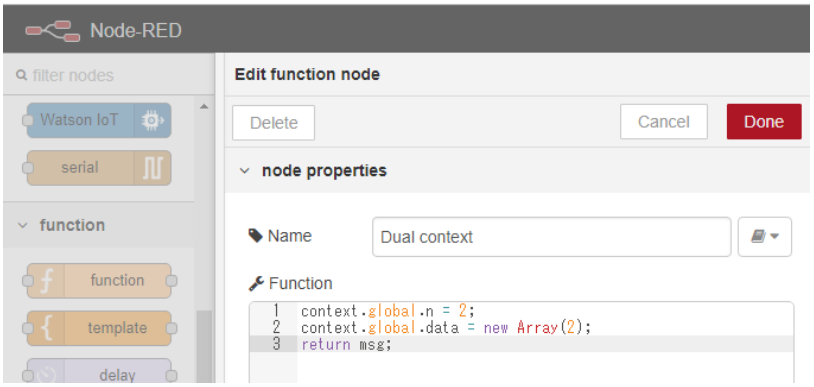


4. Drag the function node from the Function Nodes palette and drop it onto the sheet.



5. Double-click the placed function node and edit the properties of the node.

5a. Enter “Dual context” in the Name field.

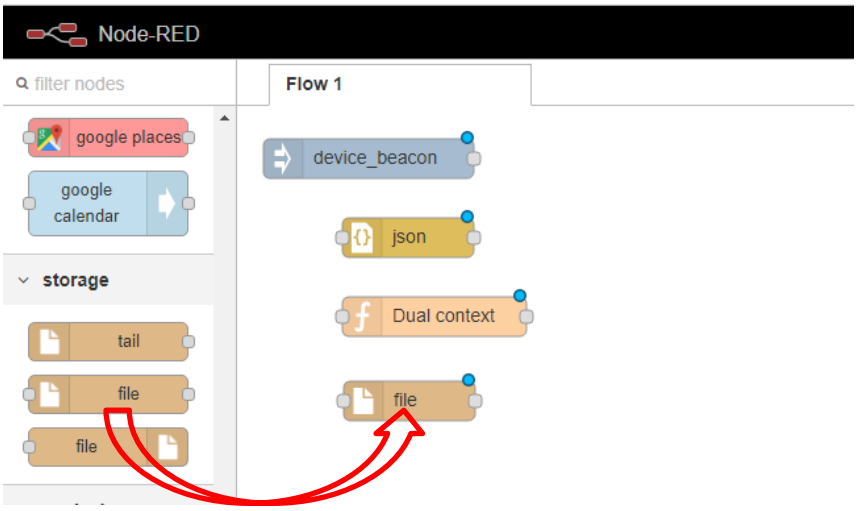


5b. Write the following to the Function filed.

```
context.global.n = 2;
context.global.data = new Array(2);
return msg;
```

5c. Click the ”Done” button.

6. Drag the file node from the Storage Nodes palette and drop it onto the sheet.

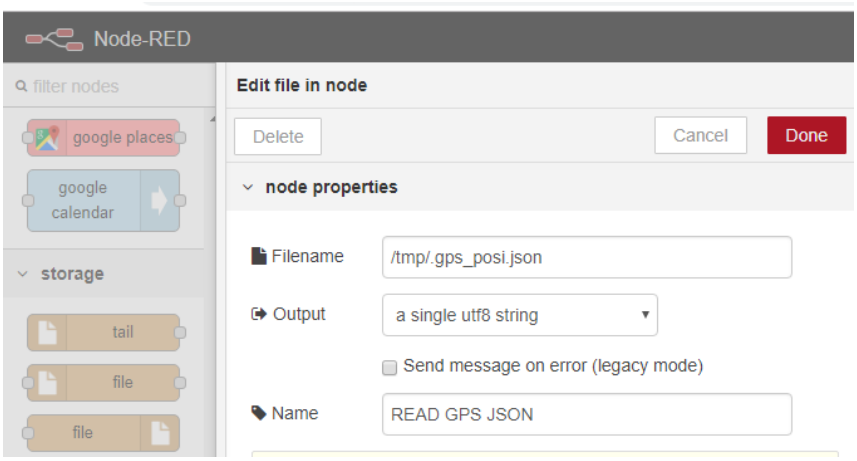


7. Double-click the placed file node and edit the properties of the node.

   7a. Enter “/tmp/.gps_posi.json” in the Fileame field.

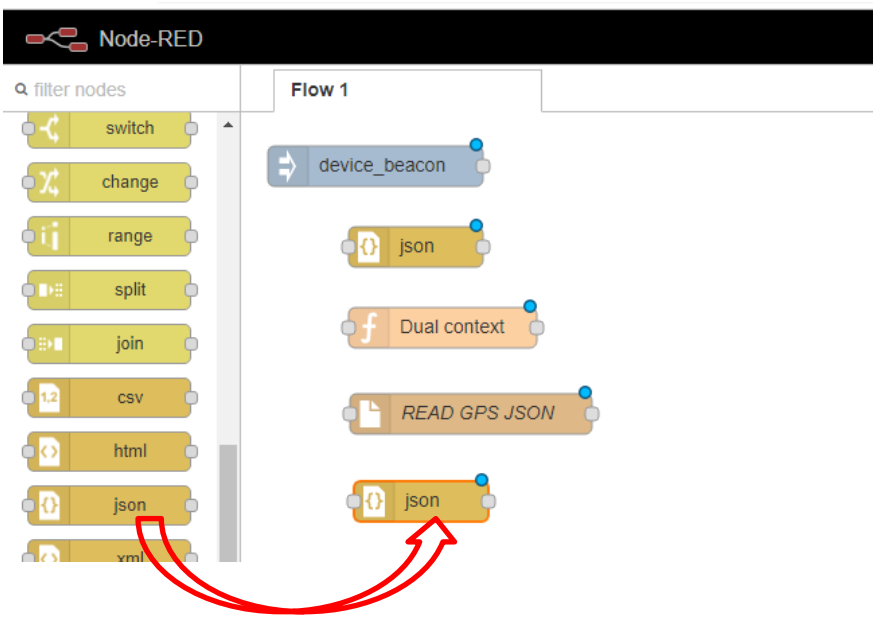
   7b. Enter “READ GPS JSON” in the Name field.

   7c. Select “a single utf8 string” in pull down menu of the Output field.

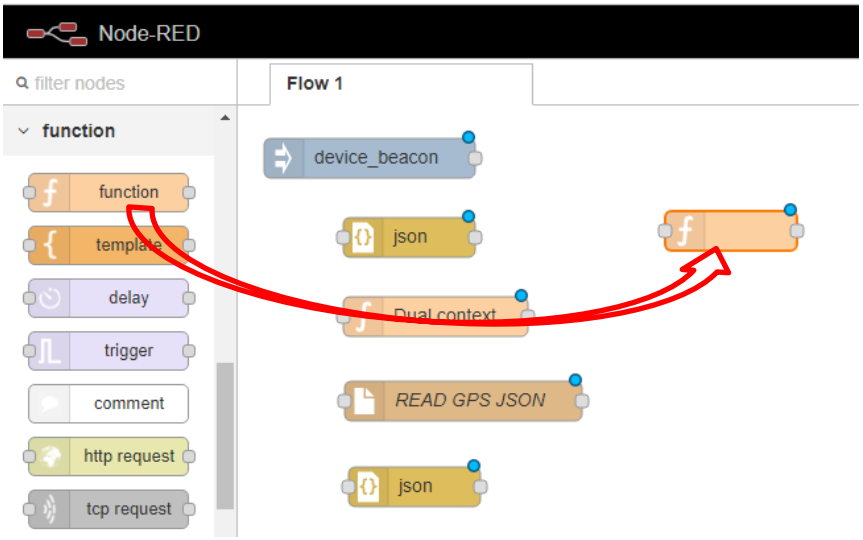


   7d. Click the ”Done” button.

8. Drag the json node from the Function Nodes palette and drop it onto the sheet.

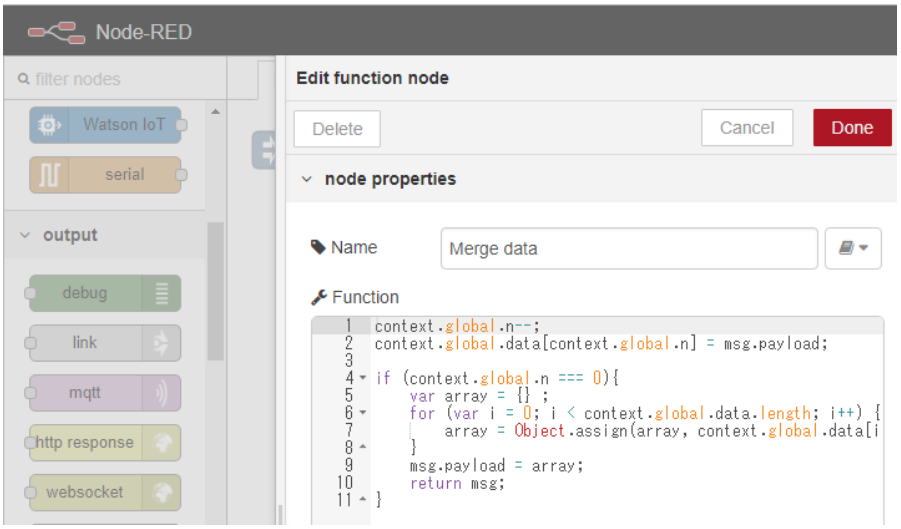


9. Drag the function node from the Function Nodes palette and drop it onto the sheet.



10. Double-click the placed function node and edit the properties of the node.

5a. Enter “Marge data” in the Name field.

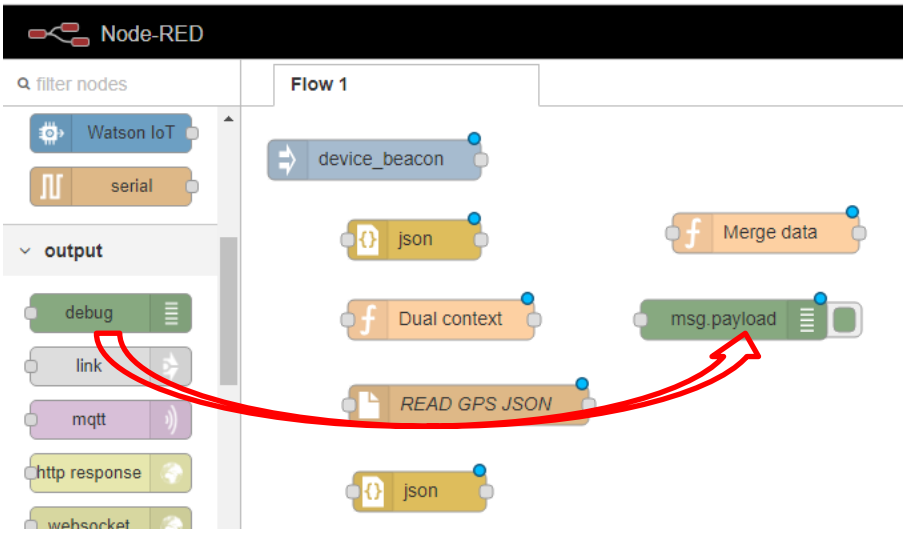


5b. Write the following to the Function filed.

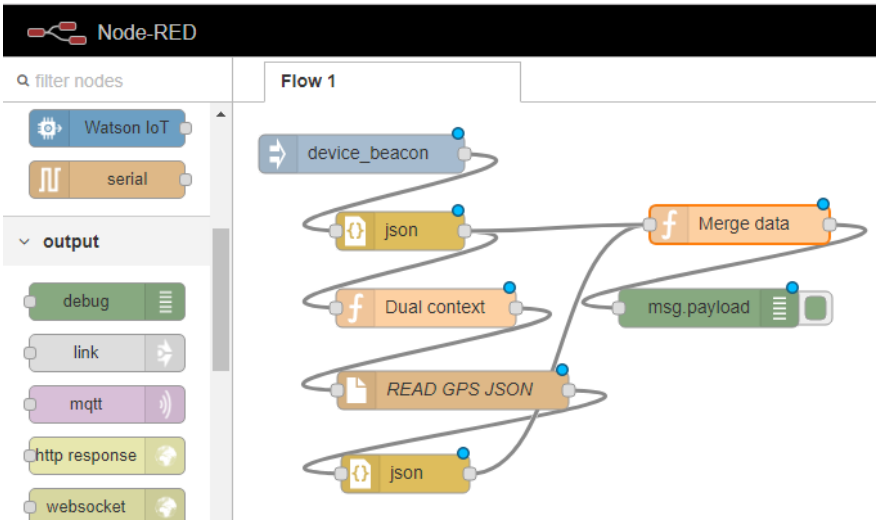
```
context.global.n--;
context.global.data[context.global.n] = msg.payload;

if (context.global.n === 0){
        var array = {} ;
        for (var i = 0; i < context.global.data.length; i++) {
                array = Object.assign(array, context.global.data[i]);
        }
        msg.payload = array;
        return msg;
}
```

7d. Click the ”Done” button.

11. To check the output result, drag the debug node from the Output Nodes palette and drop it on the sheet.



12. Connect each node as follows. After the connection is completed, click the Deploy button.

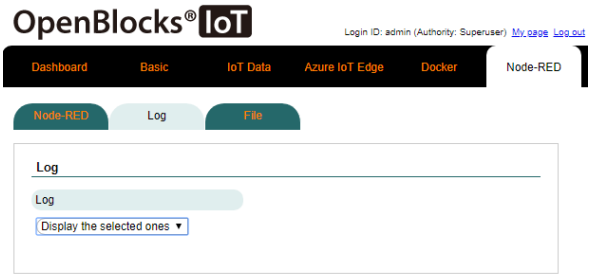


With the above, when receiving the beacon data on the Node-RED side, the global position information of the GPS information file is merged and output to the debug node.

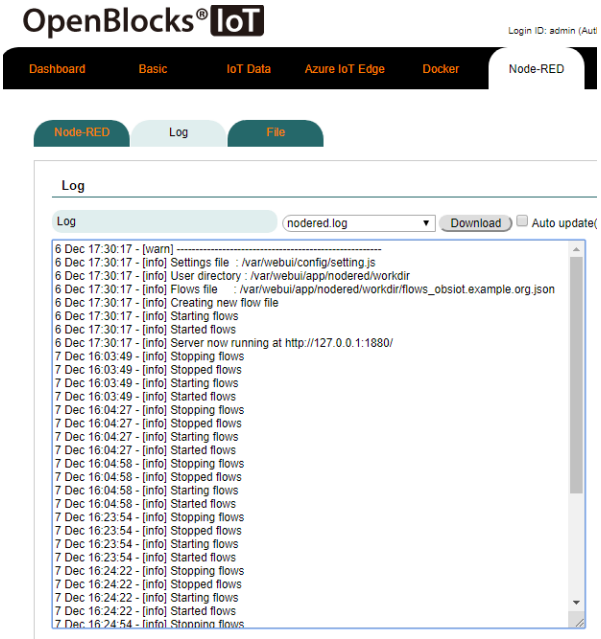
Chapter 5 Other

5-1. Node-RED log

Node-RED logs can be viewed using the **[Node – RED] –[Log]** tab.



Select the log file which want to view from the log column.



When selecting a log file, the end of the target log file is displayed.

In addition, you can download the log file by pressing the download button.

5-2. Add node to Node-RED

By uploading a file for a node it can add a node to Node - RED.

For file upload for nodes, use **[Node -RED] -[File]** tab.

Nodes file

Upload :

It can upload a single file for uploading Node-RED.

After selecting the file to be uploaded, click the Upload button.

Bulk upload :

Multiple files can be uploaded for uploading Node - RED. As a file to be uploaded, specify a file compressed in tar format.

After selecting the file to be uploaded, click the Upload button.

In order to apply the uploaded node file to Node-RED, restart of Node-RED is required.

Restart the process from the dashboard described in section 5-4.

For details on how to create Node-RED nodes, refer to the following URL.

https://nodered.jp/docs/creating-nodes/first-node

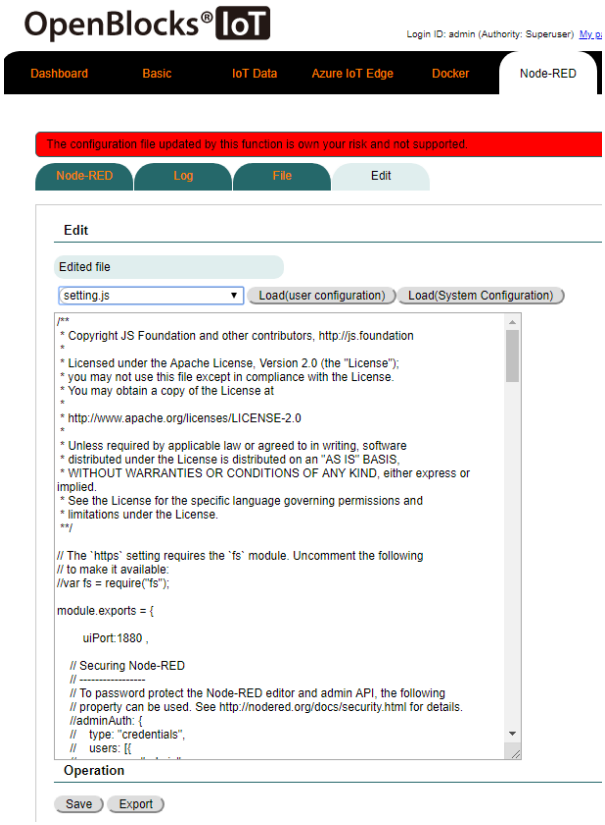
Also, in the case of node addition according to this section, creation of package.json etc. is unnecessary since nodes are not made public.

If files with incompleteness (html and js files) are uploaded, reading is skipped and the target node is not displayed.

5-3. Editing Cofiguration faile of Node-RED

If selected “User-defined configuration” in the “Use Config Settings” field of the **[Node-RED] - [Node-RED]** tab, the **[Node-RED] - [Edit]** tab is displayed.

It can edit the Node-RED configuration file using this [Node-RED]-[Edit] tab.



<u>Edit</u>

Edited file :

Please select "setting.js".

When the file is selected, the contents of the configuration file currently used are loaded.

If you want to read the settings defined by the saved user, please click the "Load(User configuration)" button.

Also, if you want to read the configuration created by the system, please press "Load(System configuration)" button.

* For editing this item, we assume that you have sufficient knowledge about Node-RED.

Click the save button to apply.

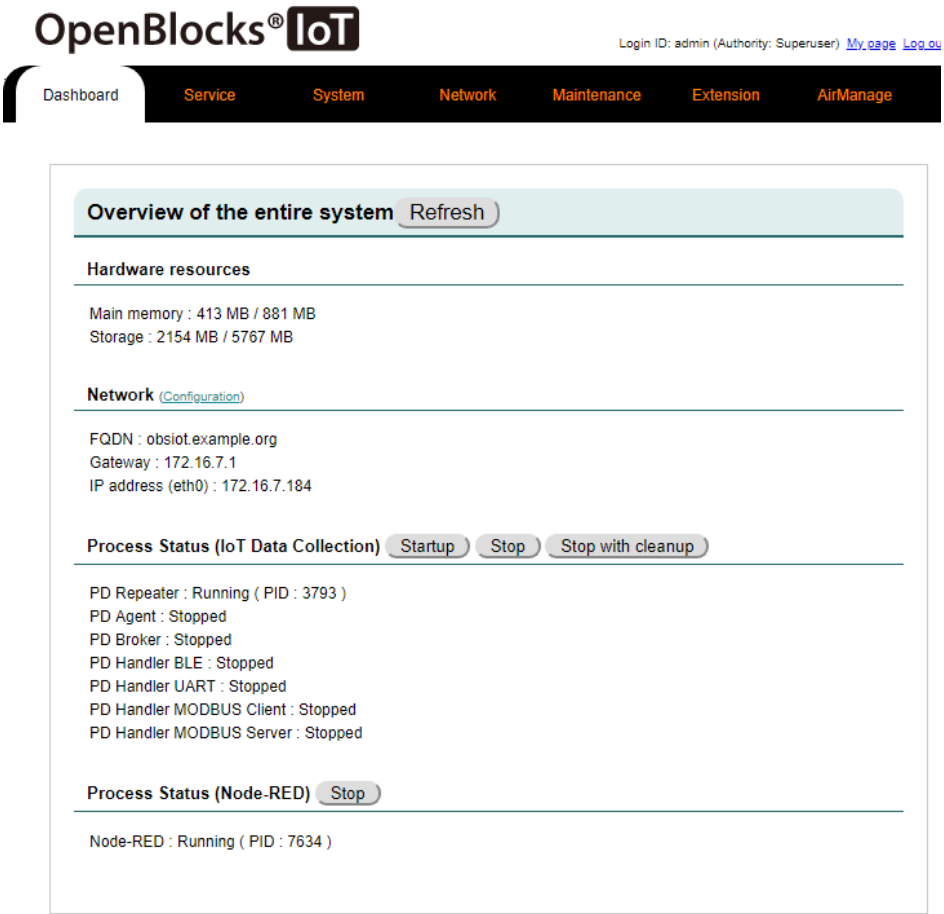
Click the export button to download configuration file which displayed, also.

5-4. Process status of Node-RED

If Node-RED is enabled, the process status of Node-RED is displayed on the [Dashboard] tab.

"Start" or "Stop" and "Initialize" buttons will be displayed that depending on the process status.

Please note that the "Initialize" button will delete the Node-RED flow.



5-5. Node-RED over HTTPS

Access to dashboard of Node-RED is usually HTTP.

If you want to use HTTPS for security issue, It can be changed by the following method.

*When changing to HTTPS, the URL link from the link button in the [Node-RED]-[Node-RED] tab will be broke.

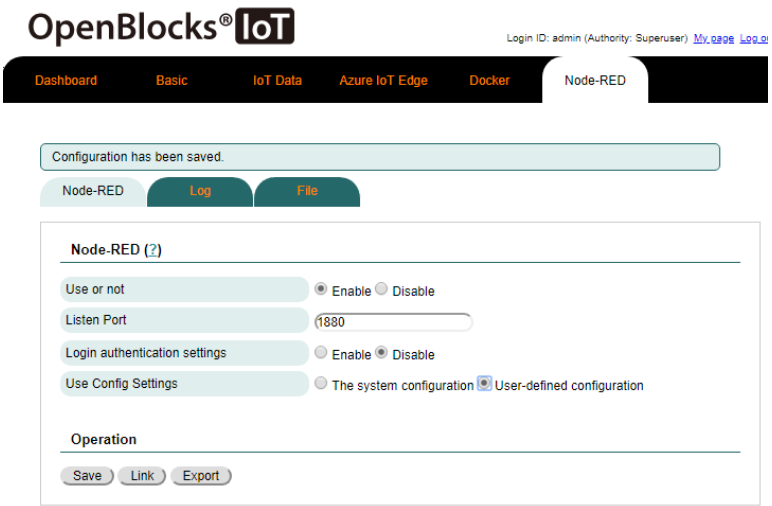
1. Create an SSL certificate

   Create a self-signed certificate under the / var / tmp directory.

   We do not recommend using a self-signed certificate for formal operation.

```
# cd /var/tmp
# openssl genrsa 2048 > privatekey.pem
# openssl req -new -key privatekey.pem 2> /dev/null > server.csr <<!
JP
Tokyo




!
# openssl x509 -days 365 -req -signkey privatekey.pem < server.csr 2> /dev/null >
certificate.pem
```

2. Selecte “User-defined configuration” in the “Use Config Settings” field of the **[Node-RED] - [Node-RED]** tab, and click the save button.



3. Modify the configuration file using the [Node-RED]-[Edit] tab.
   Select "setting.js" as the file to be edited and edit inside the text box.
   The change point is comment cancellation of the "var fs" part, comment cancellation of the https directive, and pathname modification.

*Before edit

```
                                        . . . . . .
// The `https` setting requires the `fs` module. Uncomment the following
// to make it available:
//var fs = require("fs");
                                        . . . . . .
    // The following property can be used to enable HTTPS
    // See http://nodejs.org/api/https.html#https_https_createserver_options_requestlistener
    // for details on its contents.
    // See the comment at the top of this file on how to load the `fs` module used by
    // this setting.
    //
    //https: {
    //      key: fs.readFileSync('privatekey.pem'),
    //      cert: fs.readFileSync('certificate.pem')
    //},
                                        . . . . . .
```

*After edit

```
                                        . . . . . .
// The `https` setting requires the `fs` module. Uncomment the following
// to make it available:
var fs = require("fs");
                                        . . . . . .
    // The following property can be used to enable HTTPS
    // See http://nodejs.org/api/https.html#https_https_createserver_options_requestlistener
    // for details on its contents.
    // See the comment at the top of this file on how to load the `fs` module used by
    // this setting.
    //
    https: {
        key: fs.readFileSync('/var/tmp/privatekey.pem'),
        cert: fs.readFileSync(/var/tmp/'certificate.pem')
    },
                                        . . . . . .
```

4. Click the Save button to completes the change from HTTP to HTTPS.

OpenBlocks IoT Family Node-RED Starter Guide
Version 3.3.0 (10 Dec, 2018)

Plat'Home Co., Ltd.
NIHON BUILDING KUDANBEKKAN, 3F
4-2-3, Kudankita, Chiyoda-ku, TOKYO 102-0073, JAPAN