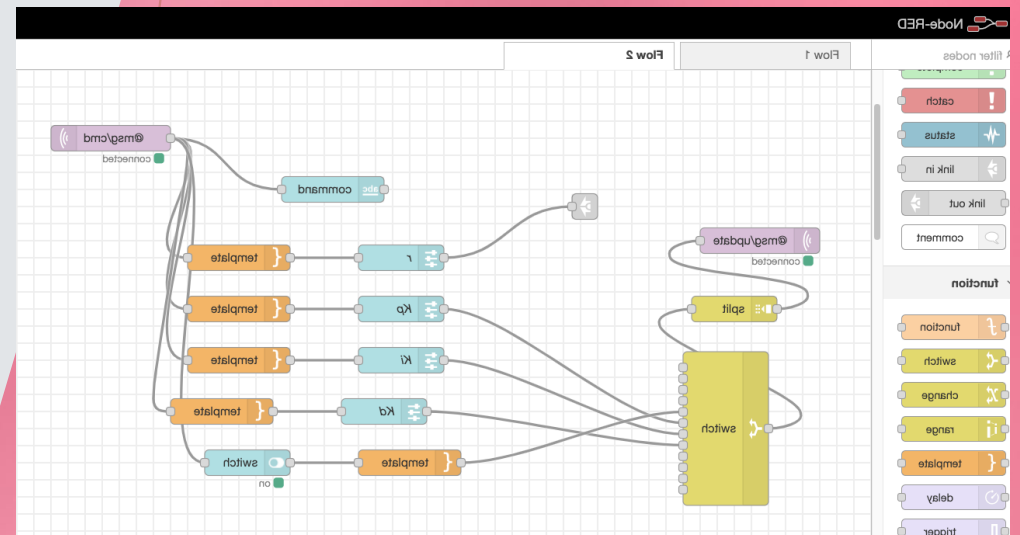


IOT WITH NODE-RED

DR.VARODOM TOOCHINDA
DEPT. OF MECHANICAL ENGINEERING
KASETSART UNIVERSITY



TOPICS

- Create device on NETPIE2020 for node-red
- MQTT in node setup
- Message from device shadow
- Use split and switch nodes to extract data from device shadow message
- Use gauge node for display
- Sending instructions via MQTT out node
- Text input, slider and switch nodes
- Data extraction from @msg/update topic to update slider and switch nodes
- Export and import node-red flows

Create new device for node-red in NETPIE portal

NETPIE

lag3

- Overview
- Device List
- Device Groups
- Freeboard
- Event Hooks
- Setting

lag3 / device / nodereddev

Information to setup mqtt-in and mqtt-out nodes

Description

Key

Client ID : cc21345d-cad2-4a0b-a806
Token : bCT8KMmnLW5E6bWm8Coe
Secret : W9L7BY69T(PMBO5*(R2Jec

Status : Offline
Enable : ☒

Shadow Schema Trigger

Tree ▼

Select a node...

- object {0}
- (empty object)

Create a group for 2 devices



lag3

Overview

Device List



Device Groups

Freeboard

Event Hooks

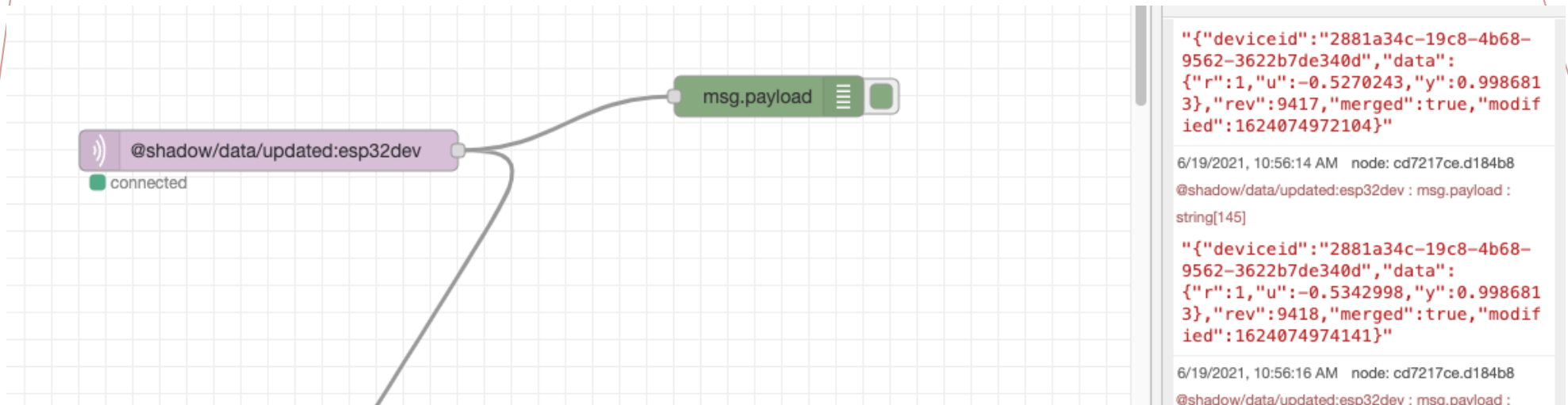
Setting

lag3 / device

<input type="checkbox"/>	Name ▾	Tags	Group ▾	Create Date ▾
<input type="checkbox"/>	 nodereddev -	-	group1	2021-06-19 10:23
<input type="checkbox"/>	 esp32dev -	-	group1	2021-06-16 12:49

1-2 of 2 item

Use MQTT in node to observe string from device shadow



MQTT in setup

Edit mqtt in node

Delete Cancel Done

Properties

Server NETPIE2020

Topic @shadow/data/updated:esp32dev (*)

QoS 2

Output auto-detect (string or buffer)

Name

(*) Set Topic to @shadow/data/updated:devicename where devicename is the device name that connects to ESP32 (not the device connected with node-red!)

Edit mqtt in node > Edit mqtt-broker node

Delete Cancel Update

Properties

Name NETPIE2020

Connection Security Messages

Server broker.netpie.io Port 1883

Use TLS

Protocol MQTT V3.1.1

Client ID cc21345d-cad2-4a0b-a806-abdc75df0c7d

Keep Alive 60

Session Use clean session

Client ID of device connected with node-red

Edit mqtt in node > Edit mqtt-broker node

Delete Cancel Update

Properties

Name NETPIE2020

Connection Security Messages

Username bCT8KMmnLW5E6bWm8Coeb4eHzzQFXo94

Password *****

Token and Secret of device connected with node-red

String pattern from shadow

6/19/2021, 10:57:31 AM node: cd7217ce.d184b8

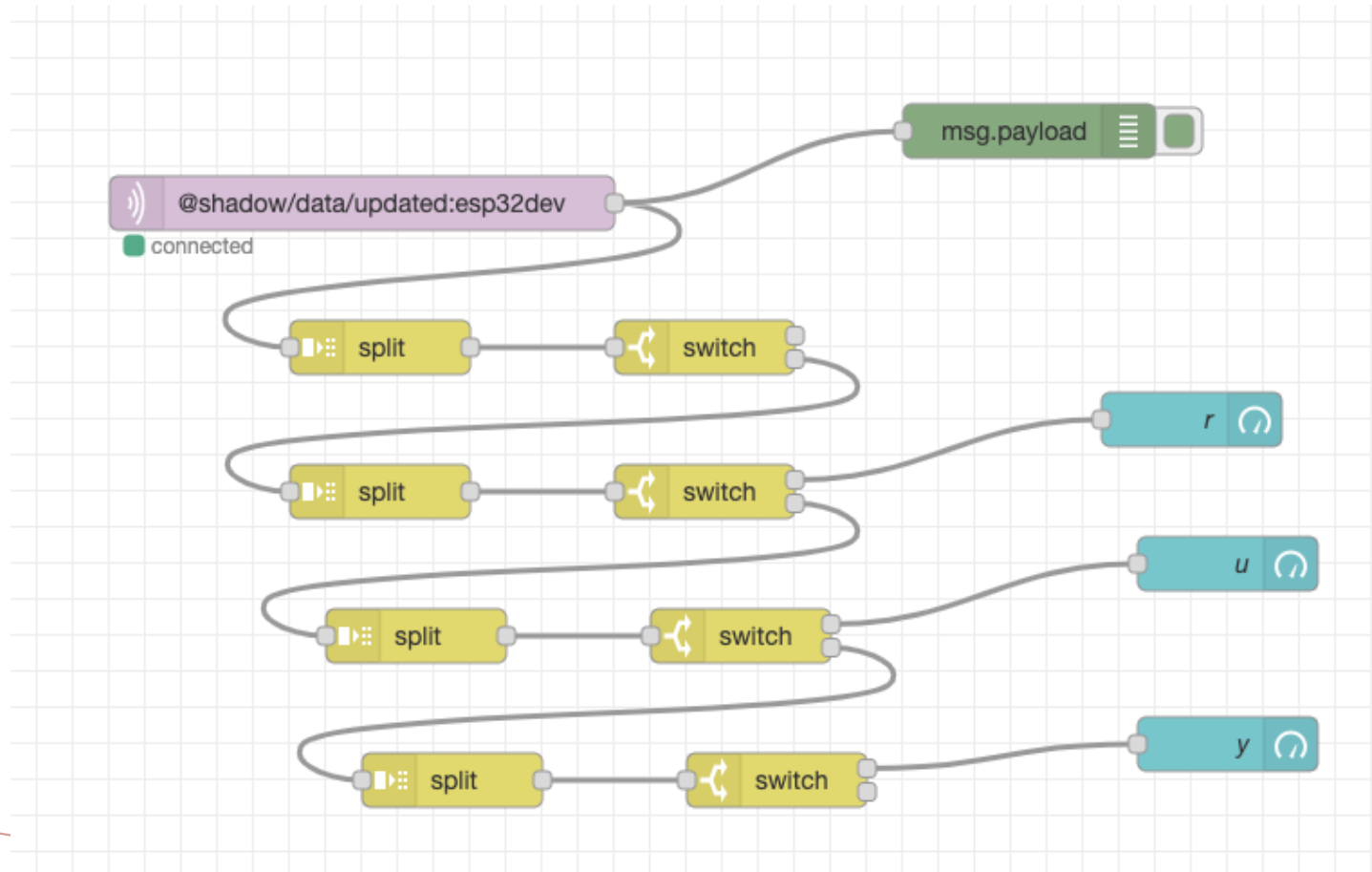
@shadow/data/updated:esp32dev : msg.payload : string[144]

```
"{"deviceid":"2881a34c-19c8-4b68-9562-3622b7de340d","data":  
{"r":1,"u":-0.6169575,"y":1.001319},"rev":9456,"merged":  
true,"modified":1624075051706}"
```

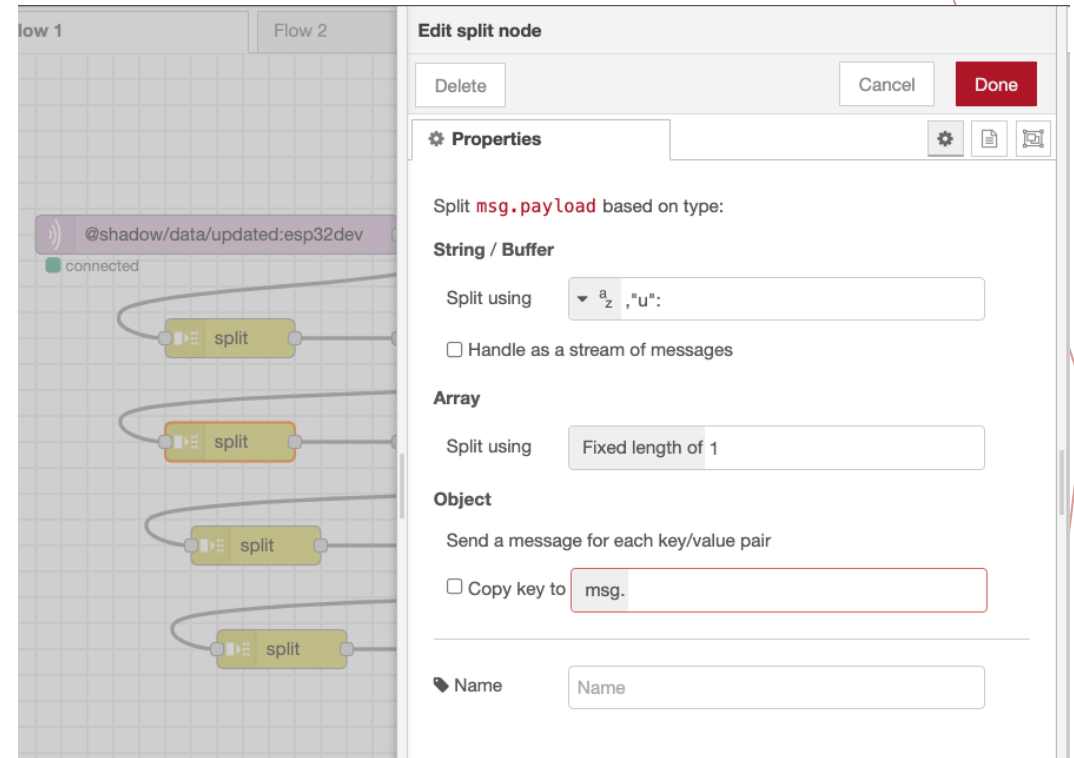
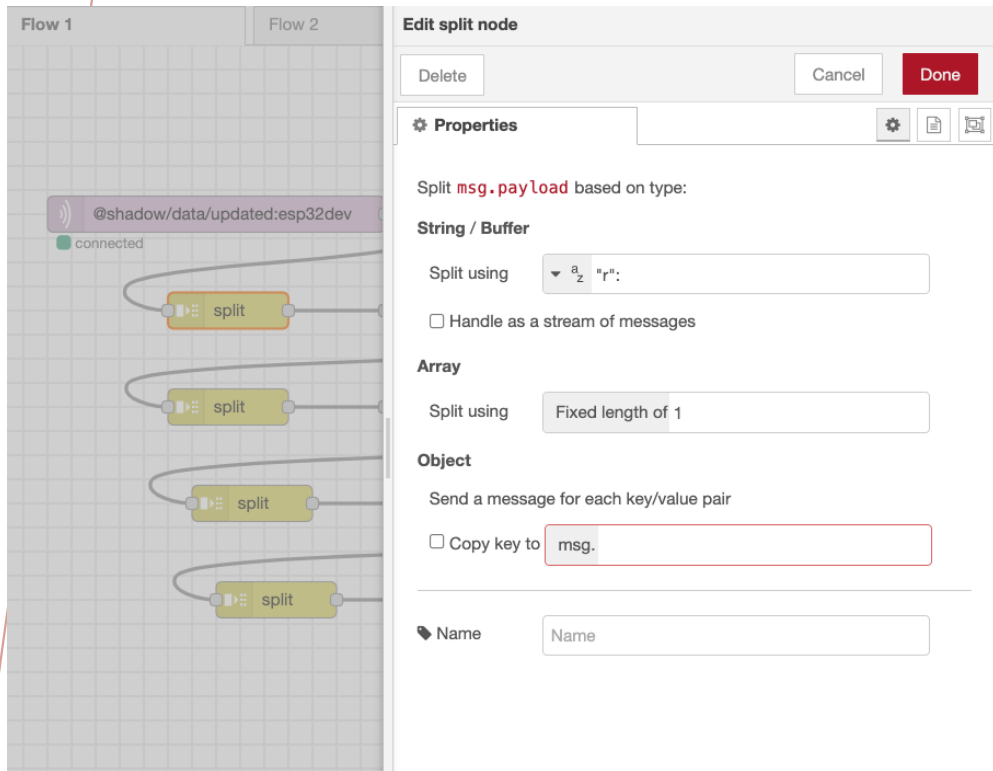


data we want to extract

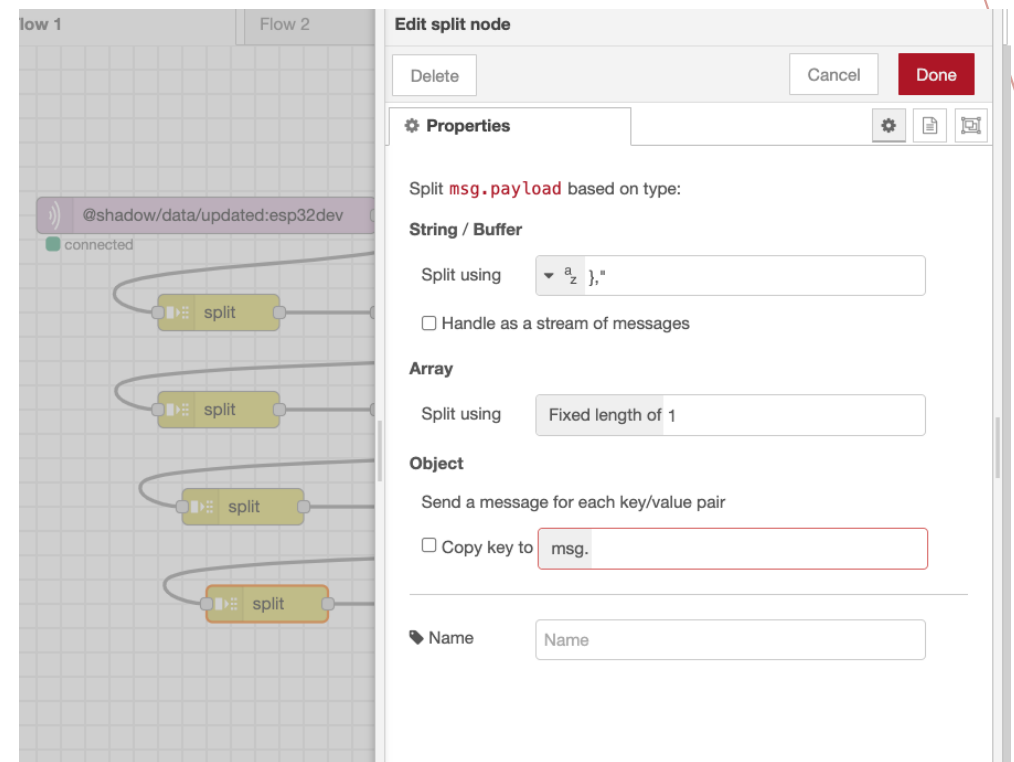
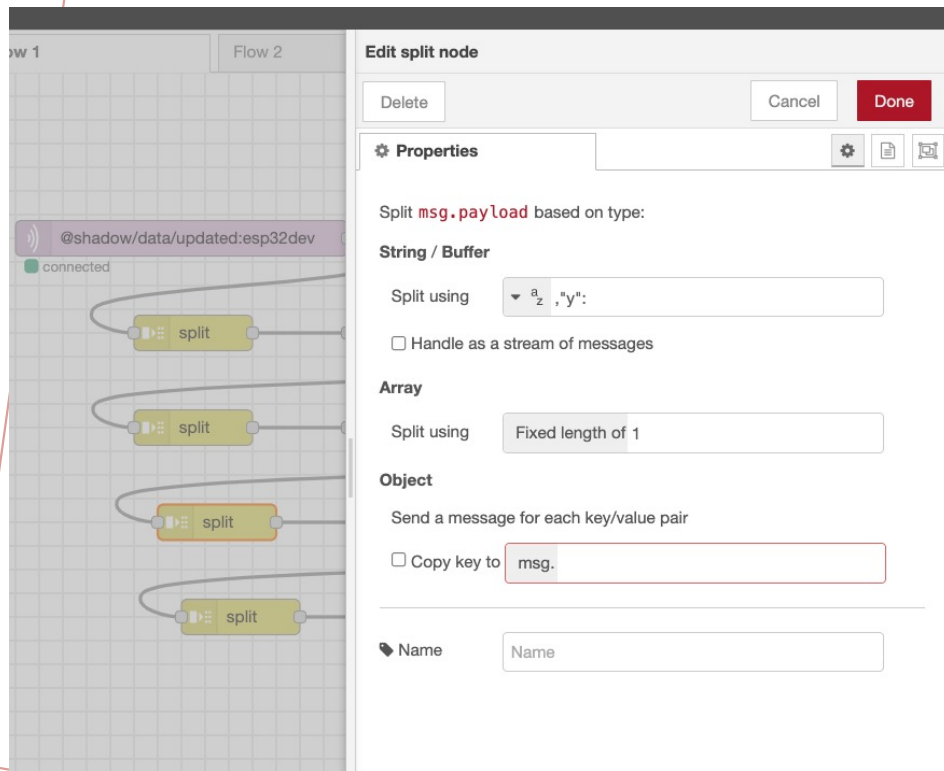
Create flow for monitoring



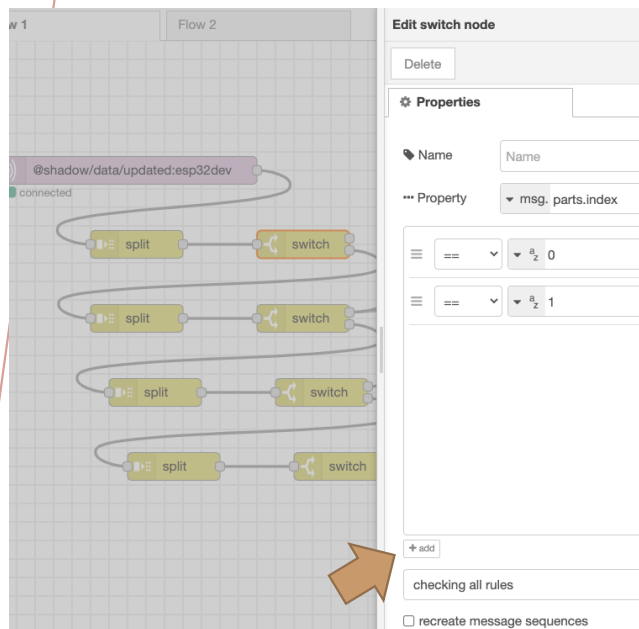
Split nodes settings (1-2)



Split nodes settings (3-4)



Switch nodes settings (similar for all 4)



เพิ่ม substring index == 1

Edit switch node

Delete Cancel Done

Properties

Name: Name

Property: msg. parts.index

== a_z 0 → 1 x

== a_z 1 → 2 x

Gauge setup example for y (the rest can be set similarly)

The image displays a software interface for configuring a gauge node. On the left, a flow diagram titled "Flow 2" shows a sequence of four "switch" blocks connected to a series of nodes labeled r , u , and y . The node y is highlighted with an orange border. On the right, the "Edit gauge node" panel is open, showing the configuration for the selected node.

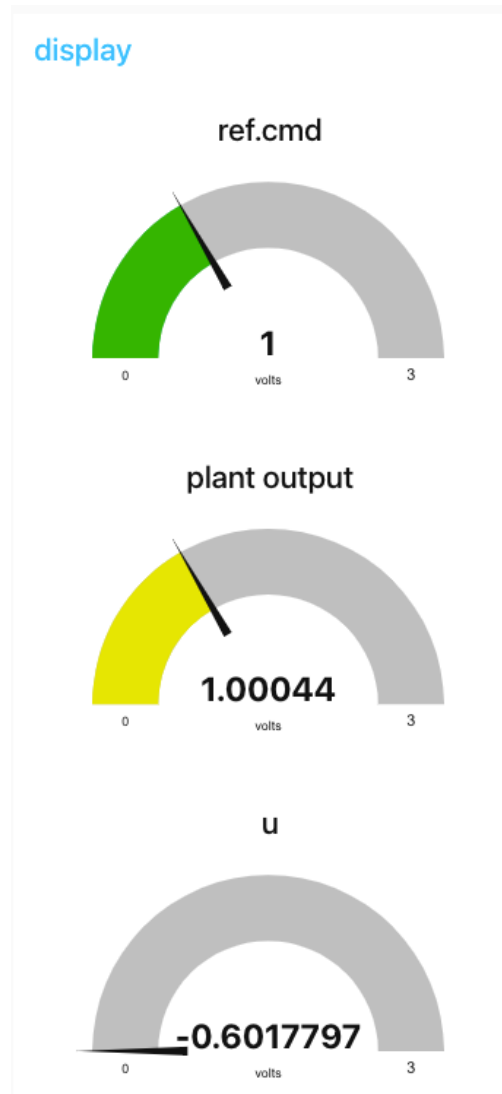
Edit gauge node

Buttons: Delete, Cancel, Done

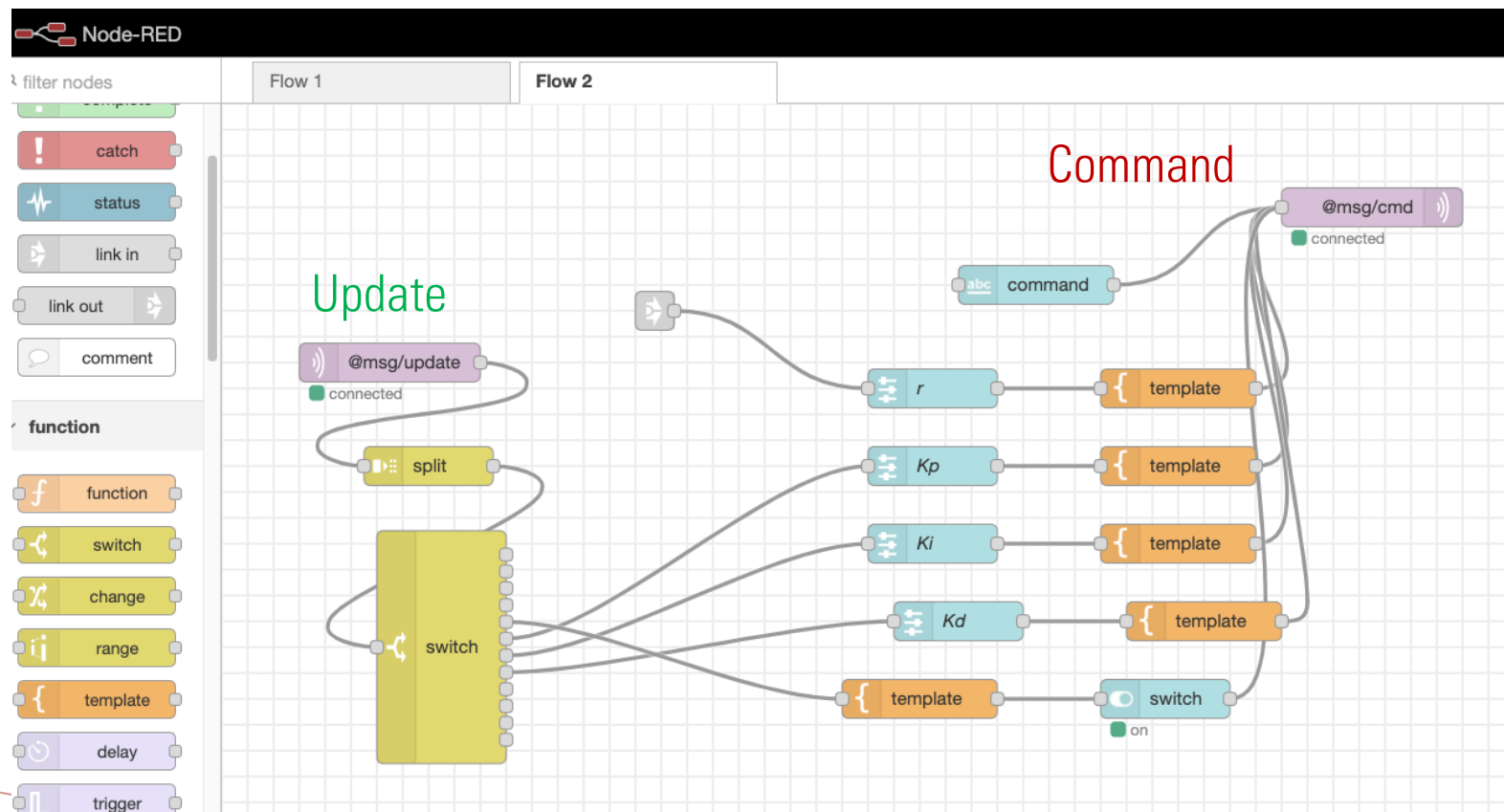
Properties

- Group: [lag3] display
- Size: auto
- Type: Gauge
- Label: plant output
- Value format: {{value}}
- Units: volts
- Range: min 0, max 3
- Colour gradient: [Green, Yellow, Red]
- Sectors: 0, ..., 1, ..., 2, ..., 3
- Name: y

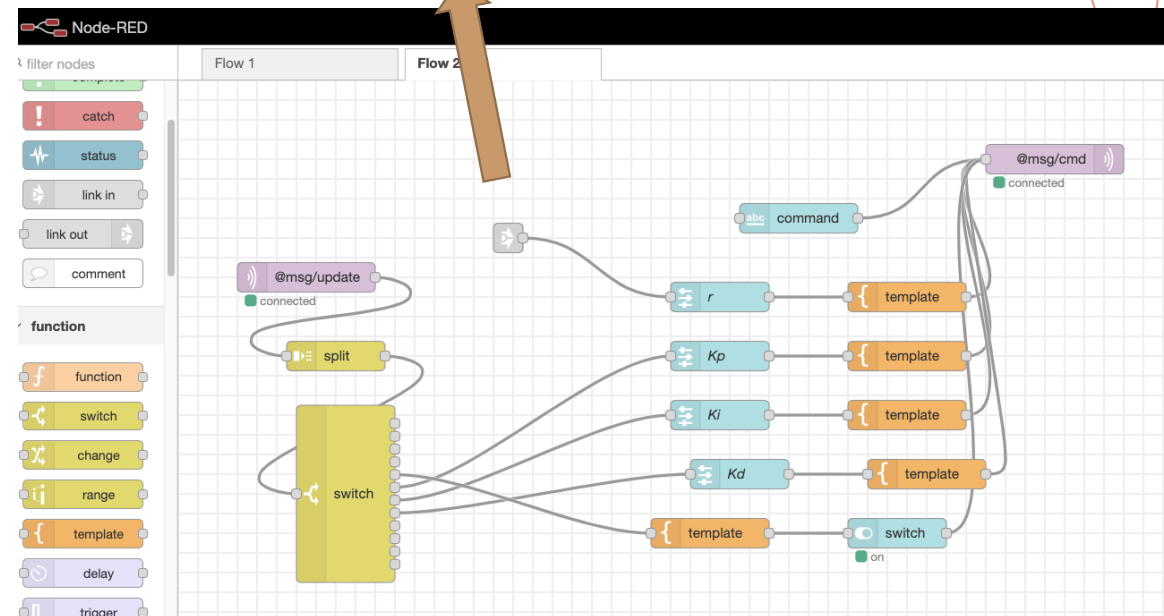
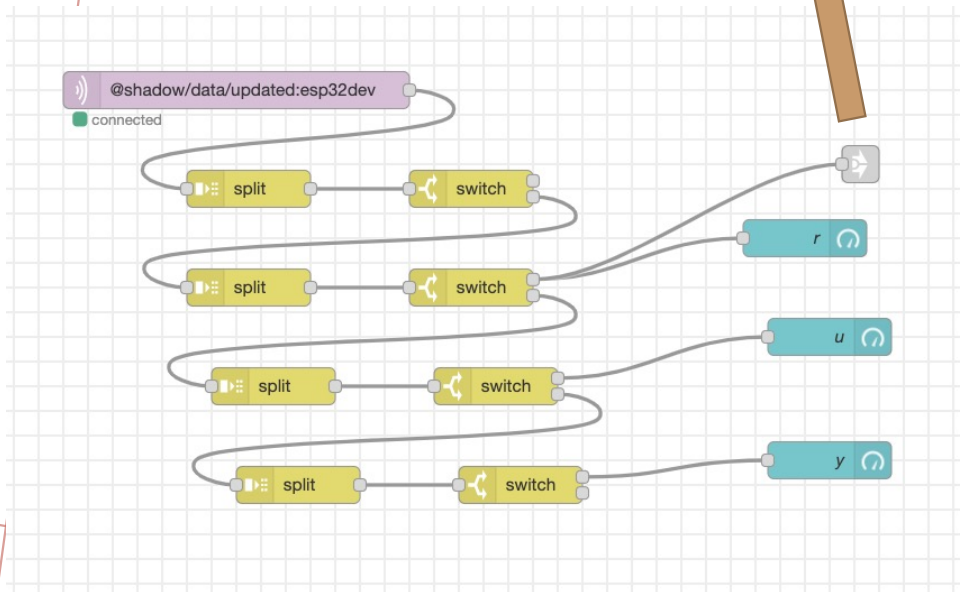
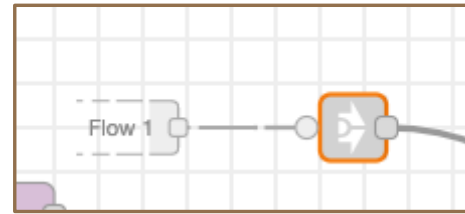
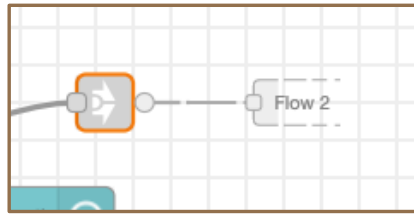
Dashboard *(display section)*



Create flow for command and update



flow connection with link in, link out nodes



Topic setup for MQTT node

Edit mqtt in node

Delete Cancel Done

Properties

Server NETPIE2020

Topic @msg/update

QoS 2

Output auto-detect (string or buffer)

Name Name

Edit mqtt out node

Delete Cancel Done

Properties

Server NETPIE2020

Topic @msg/cmd

QoS Retain

Name Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Text Input setup

Edit text input node

Delete Cancel Done

Properties

Group [lag3] cmd

Size auto

Label command

Tooltip optional tooltip

Mode text input Delay (ms) 0

→ If **msg** arrives on input, pass through to output: ☒

When changed, send:

Payload	Current value
Topic	msg. topic
Name	

Setting **Delay** to 0 waits for Enter or Tab key, to send input.

kp slider and template setup example

Edit slider node

Delete Cancel Done

Properties

Group [lag3] cmd

Size auto

Label Kp

Tooltip optional tooltip

Range min 0 max 10 step 0.1

Output only on release

☒ If msg arrives on input, pass through to output:

☒ When changed, send:

Payload Current value

Topic msg. topic

Name Kp

Edit template node

Delete Cancel Done

Properties

Name Name

Property msg. payload

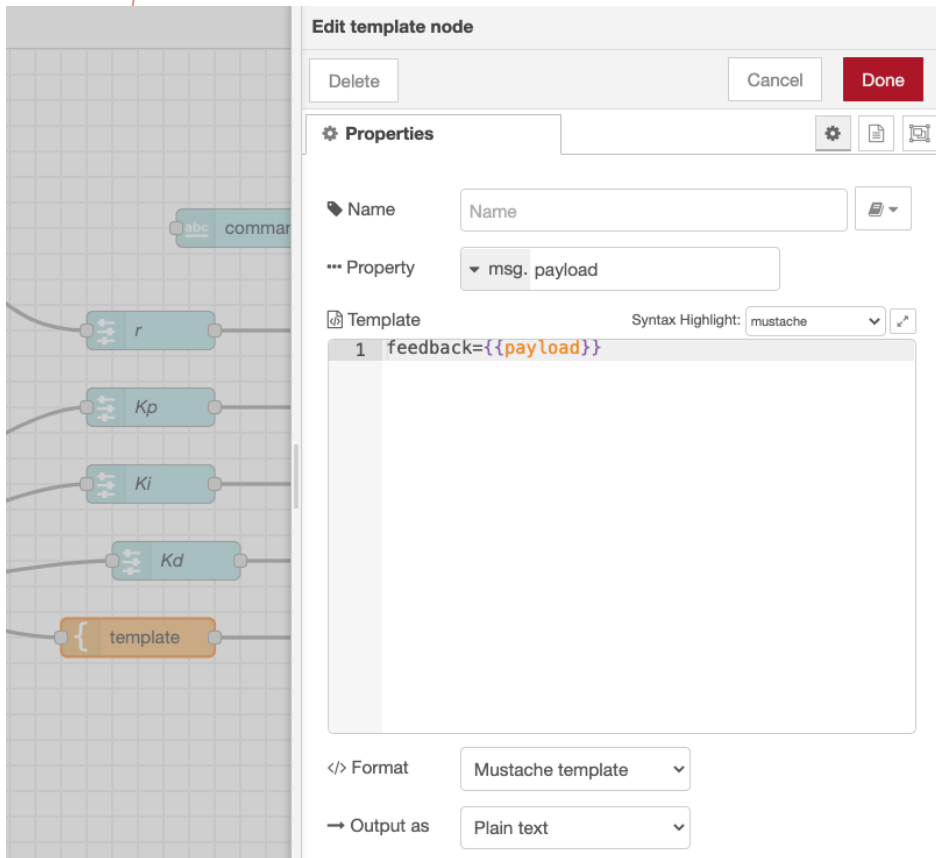
Template Syntax Highlight: mustache

```
1 kp={{payload}}
```

Format Mustache template

Output as Plain text

template and switch dashboard



The screenshot shows the 'Edit template node' dialog box. On the left, a partial view of a dashboard grid is visible, containing nodes labeled 'abc', 'command', 'r', 'Kp', 'Ki', 'Kd', and 'template'. The dialog has a title bar 'Edit template node' with 'Delete', 'Cancel', and 'Done' buttons. The 'Properties' section includes a 'Name' field, a 'Property' dropdown set to 'msg. payload', and a 'Template' text area with the content '1 feedback={{payload}}'. The 'Syntax Highlight' dropdown is set to 'mustache'. At the bottom, the 'Format' dropdown is set to 'Mustache template' and the 'Output as' dropdown is set to 'Plain text'.

Edit template node

Delete Cancel Done

Properties

Name

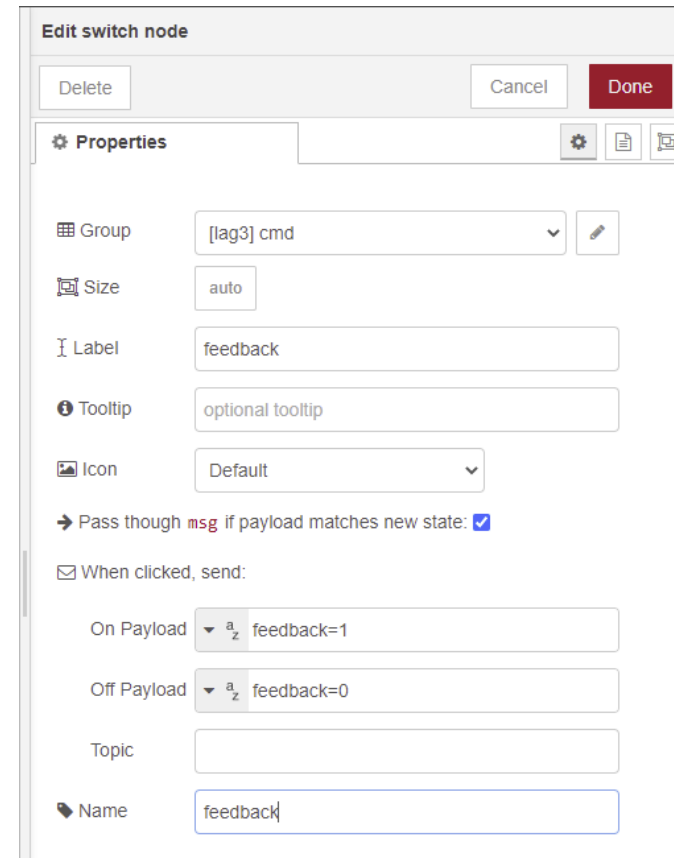
Property msg. payload

Template Syntax Highlight: mustache

```
1 feedback={{payload}}
```

Format Mustache template

Output as Plain text



The screenshot shows the 'Edit switch node' dialog box. The title bar is 'Edit switch node' with 'Delete', 'Cancel', and 'Done' buttons. The 'Properties' section includes a 'Group' dropdown set to '[lag3] cmd', a 'Size' dropdown set to 'auto', a 'Label' field set to 'feedback', a 'Tooltip' field set to 'optional tooltip', and an 'Icon' dropdown set to 'Default'. There is a checkbox 'Pass though msg if payload matches new state:' which is checked. Below this is a section 'When clicked, send:' with 'On Payload' set to 'a_z feedback=1' and 'Off Payload' set to 'a_z feedback=0'. The 'Topic' field is empty, and the 'Name' field is set to 'feedback'.

Edit switch node

Delete Cancel Done

Properties

Group [lag3] cmd

Size auto

Label feedback

Tooltip optional tooltip

Icon Default

Pass though msg if payload matches new state: ☒

When clicked, send:

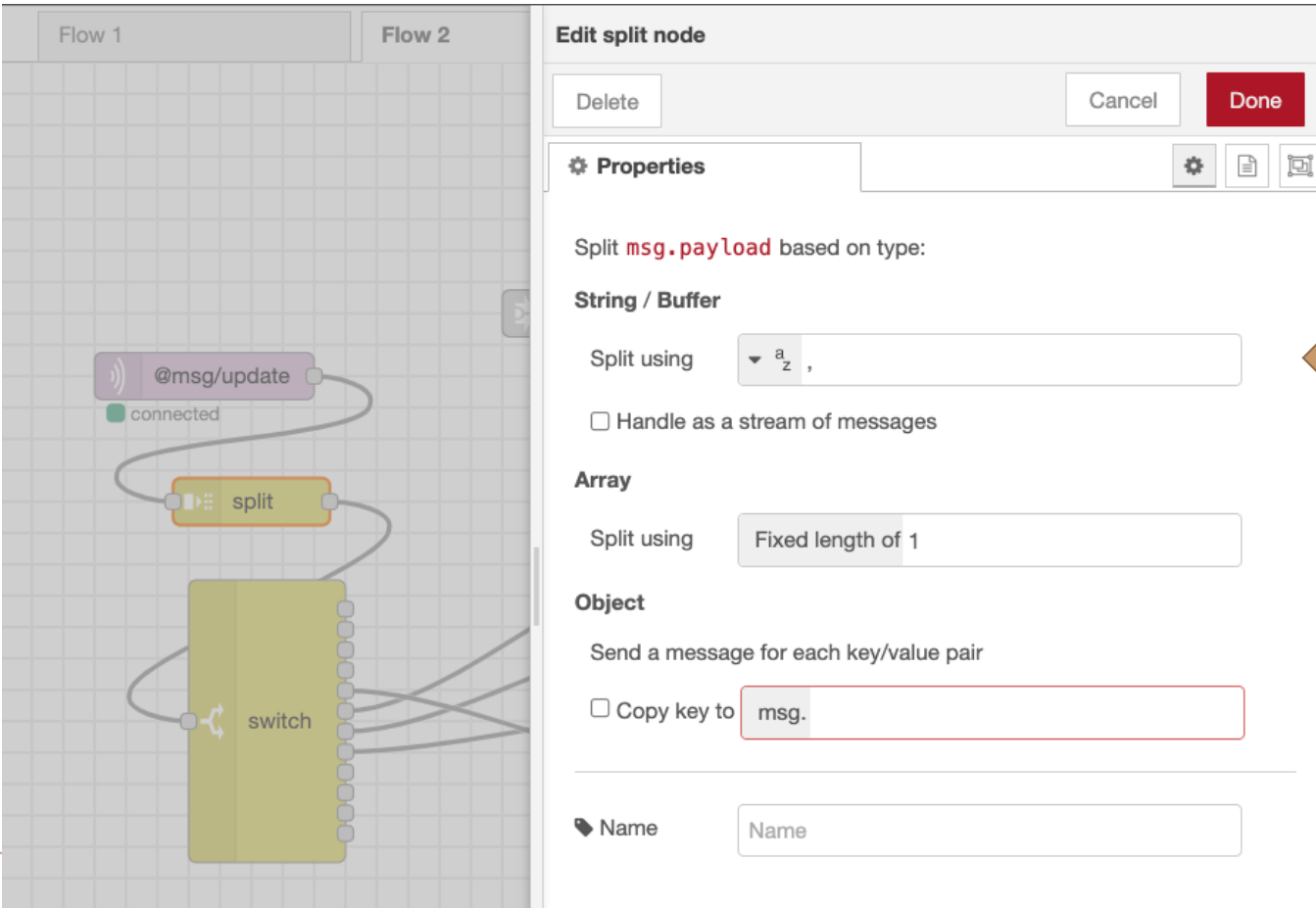
On Payload a_z feedback=1

Off Payload a_z feedback=0

Topic

Name feedback

split node setup for dashboard update



The image shows a Node-RED interface with two flows. Flow 1 contains a purple '@msg/update' node connected to an orange 'split' node, which is then connected to a green 'switch' node. Flow 2 is empty. The 'Edit split node' configuration panel is open on the right. It has tabs for 'Delete', 'Cancel', and 'Done'. The 'Properties' tab is active, showing options to split the message based on type. The 'String / Buffer' section is selected, with 'Split using' set to 'a-z ,'. A brown arrow points to this field. The 'Array' section is also visible, with 'Split using' set to 'Fixed length of 1'. The 'Object' section has 'Send a message for each key/value pair' checked and 'Copy key to' set to 'msg.'. The 'Name' field is empty.

Flow 1 Flow 2

@msg/update
connected

split

switch

Edit split node

Delete Cancel Done

Properties

Split **msg.payload** based on type:

String / Buffer

Split using ←

☐ Handle as a stream of messages

Array

Split using

Object

Send a message for each key/value pair

☐ Copy key to

Name

*switch node
setup for
dashboard
update*

Total output = 12 substrings

Flow 1

Edit switch node

Delete Cancel Done

Properties

Name Name

Property msg. parts.index

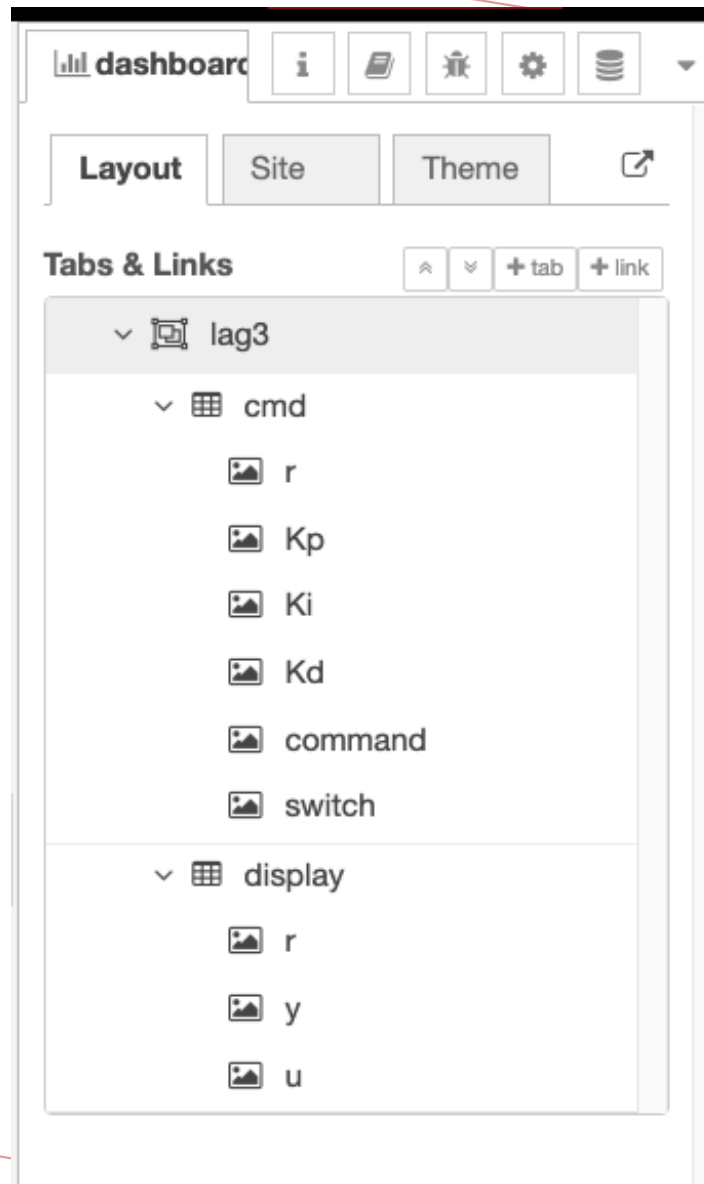
==	a_z	0	→ 1	x
==	a_z	1	→ 2	x
==	a_z	2	→ 3	x
==	a_z	3	→ 4	x
==	a_z	4	→ 5	x
==	a_z	5	→ 6	x

+ add

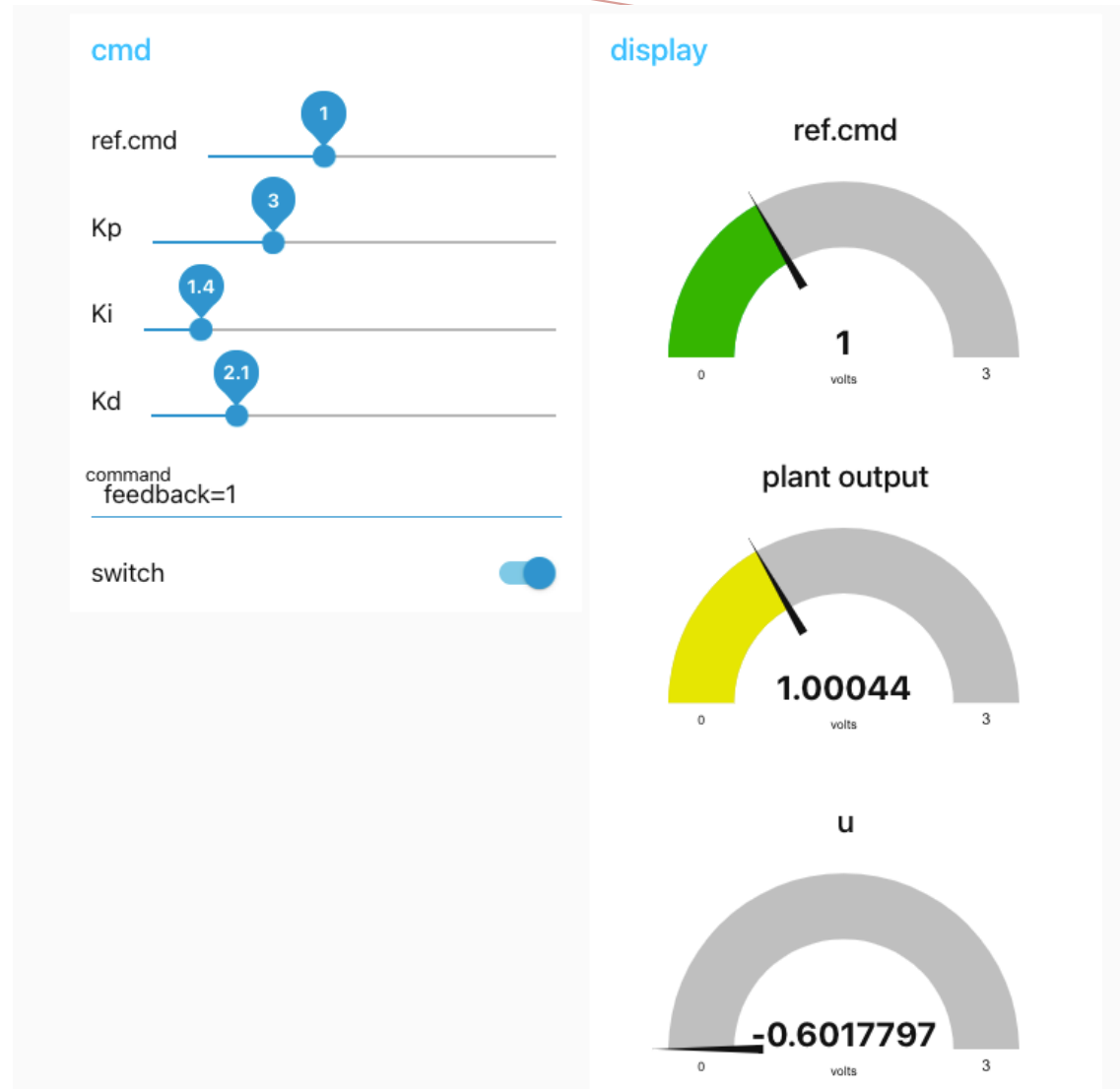
checking all rules

☐ recreate message sequences

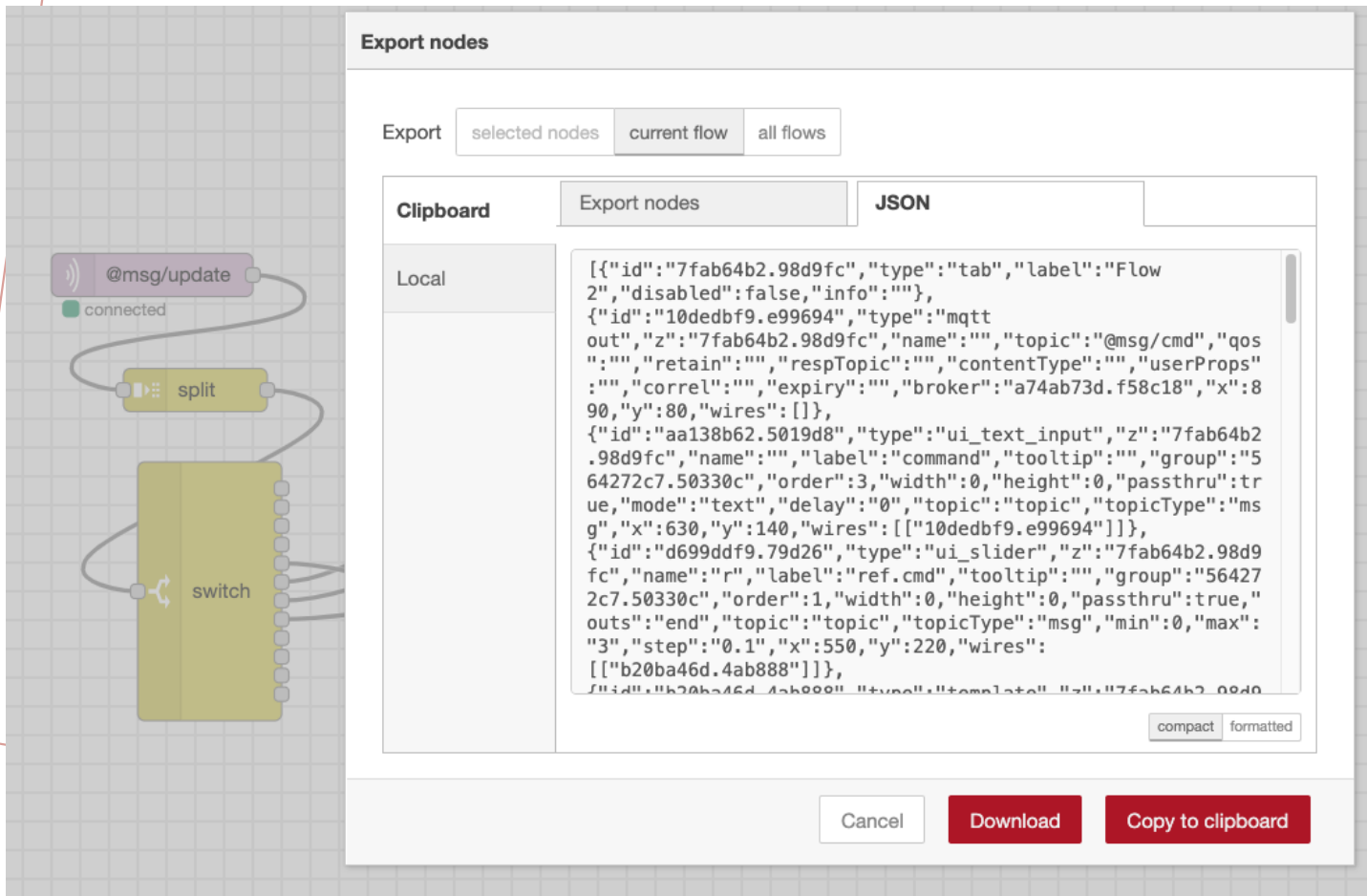
Organize dashboard



Dashboard cmd section



Export flow



The screenshot shows the Node-RED web interface. On the left, a flow is visible with three nodes: '@msg/update' (purple), 'split' (yellow), and 'switch' (yellow). The '@msg/update' node is marked as 'connected'. An 'Export nodes' dialog box is open in the center. The dialog has three tabs: 'selected nodes', 'current flow', and 'all flows'. The 'selected nodes' tab is active. Below the tabs, there are three sub-tabs: 'Clipboard', 'Export nodes', and 'JSON'. The 'JSON' sub-tab is active, displaying a large block of JSON code. At the bottom of the dialog, there are three buttons: 'Cancel', 'Download', and 'Copy to clipboard'.

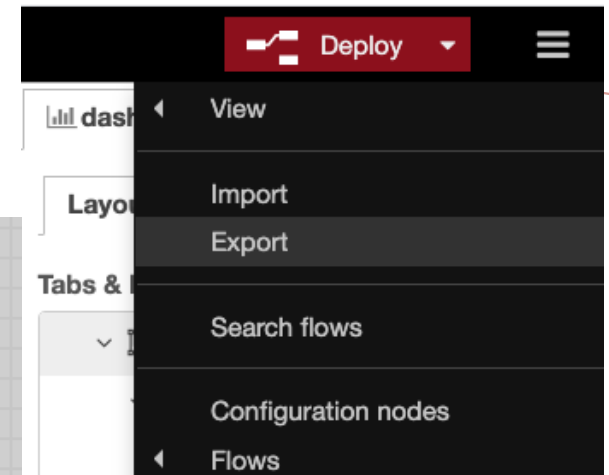
Export nodes

Export: ☐ selected nodes ☐ current flow ☐ all flows

Clipboard **Export nodes** **JSON**

Local

```
[{"id":"7fab64b2.98d9fc","type":"tab","label":"Flow 2","disabled":false,"info":""}, {"id":"10dedbf9.e99694","type":"mqtt out","z":"7fab64b2.98d9fc","name":"","topic":"@msg/cmd","qos":"","retain":"","respTopic":"","contentType":"","userProps":"","correl":"","expiry":"","broker":"a74ab73d.f58c18","x":890,"y":80,"wires":[]}, {"id":"aa138b62.5019d8","type":"ui_text_input","z":"7fab64b2.98d9fc","name":"","label":"command","tooltip":"","group":"564272c7.50330c","order":3,"width":0,"height":0,"passthru":true,"mode":"text","delay":0,"topic":"topic","topicType":"msg","x":630,"y":140,"wires":[["10dedbf9.e99694"]]}, {"id":"d699ddf9.79d26","type":"ui_slider","z":"7fab64b2.98d9fc","name":"r","label":"ref.cmd","tooltip":"","group":"564272c7.50330c","order":1,"width":0,"height":0,"passthru":true,"outs":"end","topic":"topic","topicType":"msg","min":0,"max":3,"step":0.1,"x":550,"y":220,"wires":[["b20ba46d.4ab888"]]}, {"id":"b20ba46d.4ab888","type":"template","z":"7fab64b2.98d9fc","name":"","label":"","tooltip":"","group":"","order":2,"width":0,"height":0,"passthru":true,"mode":"text","delay":0,"topic":"topic","topicType":"msg","x":550,"y":220,"wires":[]}]
```



The screenshot shows the top navigation bar of the Node-RED interface. It includes a 'Deploy' button with a refresh icon, a hamburger menu icon, and a dropdown menu. The dropdown menu is open, showing options: 'View', 'Import', 'Export', 'Search flows', 'Configuration nodes', and 'Flows'. The 'Export' option is highlighted.

- View
- Import
- Export
- Search flows
- Configuration nodes
- Flows