IBM I + node-RED =
succeed *faster*
in programming

Who are we?

- Small IBM i ISV and IBM business partner located in Oostkamp, (near Bruges) Belgium.
- Working with IBM i and its predecessors for more than 40 years.
- Applications : accountancy, real estate and customs.
- Expertise in RPG, SQL, PHP, HTML, Unity, nodejs, linux…
- Website : www.cdinvest.be
- IBM Champion 2018/2019 and IBM Fresh Face 2017
- **What you don't know how to do, we do.**

Case studies

- JORI : https://www.ibm.com/case-studies/jori
- Fibrocit : https://www.ibm.com/case-studies/fibrocit-systems-furniture-design
- Cras : https://www.ibm.com/case-studies/cras-systems-open-source
- Oris : https://www.ibm.com/case-studies/ORIS
- Deknudt Frames : https://www.ibm.com/case-studies/deknudt-frames
- Bonehill : https://www.ibm.com/case-studies/immo-bonehill-systems-hardware-website-compliance
- Vanmaele : https://www.ibm.com/case-studies/wijnen-van-maele-systems-software-ibm-i
- Winsol : https://www.ibm.com/case-studies/winsol-systems-hardware-manufacturing-digitization

Agenda

- what is node-RED ?
- first flows
- visual recognition app
- dashboard app
- chatbot app
- db2 integration + AI analysis

Agenda

- real life use of node-RED
- deep learning – what is it ?
- geomarketing – deep learning example
- next steps

Why Node-RED

Ever had one of those days…

Where the Application works!

And then…

- Can we also get some data from the this whatchamacallit?
- And send the logs off to this other server…
- And add some additional REST endpoints…

Why Node-RED - The Problem

We don't have a simple tool for co-ordinating Events, Business events – status of processes, alerts from machines Social events – tweets, alerts, Internet of Things events – temperatures, weather, lights, doors, …

Something that anyone can use to build situational applications

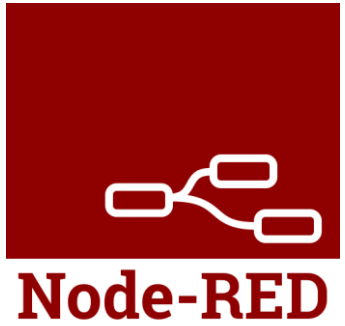
“Wouldn't it be neat if, when x happens it can tell me…

… and alert Fred…

… and kick off the xyz process…

… or just go ping !”

Why Node-RED



- The internet does not have a one-size-fits-all solution
- Every new "thing" has a new API that must be understood
- Solutions often require pulling together several different device API's and online services in new and interesting ways
- Time spent pondering how to access a serial port, or complete an OAuth flow to Twitter is not time spent on creating the real time of a solution

Node-RED =

- An application composition tool experience
- A lightweight proof of concept runtime
- Easy to use for simple tasks
- Simple to extend to add new capabilities and types of integration
- Capable of creating the back-end glue between social applications and business applications
- A great way to try…

  "can I just get this data from here to there?"

  "and maybe change it just slightly along the way…"

Node-RED ≠

- A fully-scalable, high-performance, enterprise-capable application runtime
- A dashboard with widgets
- A mobile application builder
- …

On the other hand …

- Node-RED is deployed in a manufacturing production line
- Node-RED can deployed on IBM i glue applications together
- Can be used to --quickly-- build a proof of concept
- Runs on Raspberry Pi's + Arduinos + Sensors
- Sensor readings and initial processing coordinated via Node-RED
- Able to adapt and change quickly – redeploying during support phone calls!

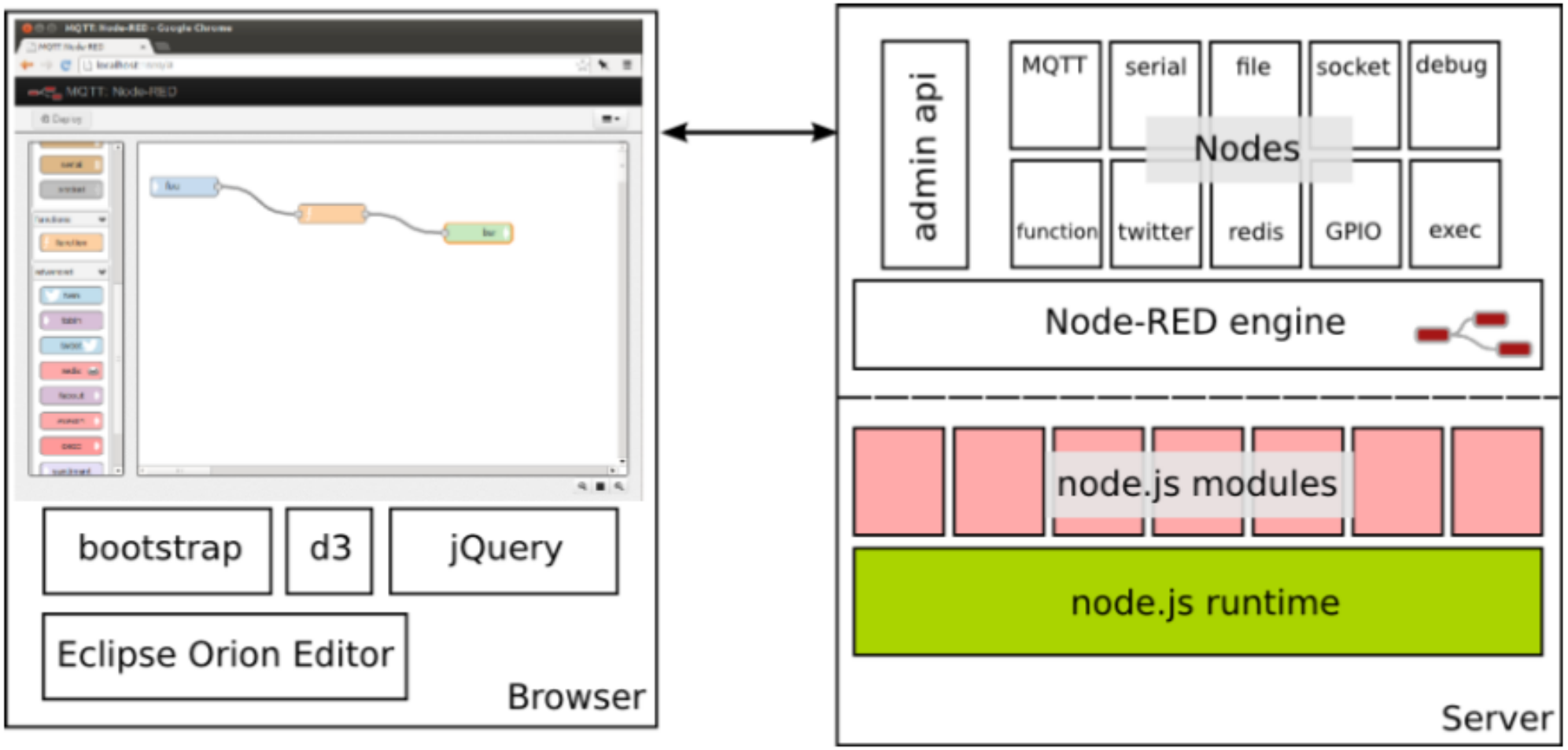
Architecture of Node-RED

- **Node.js v8-engine driven;**
  so it's fast and can use the 29000+ open-source npm modules…
- **Event-driven, asynchronous io;**
  it's all about the events
- **Single-threaded eventqueue;**
  built for fairness
- **Javascript front and back;**
  only one language runtime to deal with
- Built using express, d3, jquery and ws

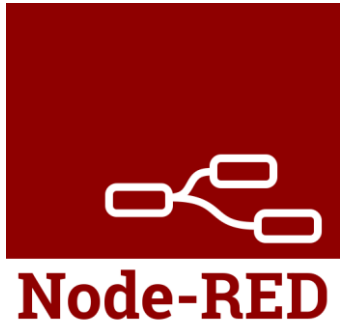
Architecture of Node-RED

- Node-RED nodes provide integration with other systems. Each node is defined in their own pair of JavaScript and html files using a simple API and are dynamically loaded by the engine.
- Web interface can be secured or run headless;

Architecture of Node-RED



Node-RED



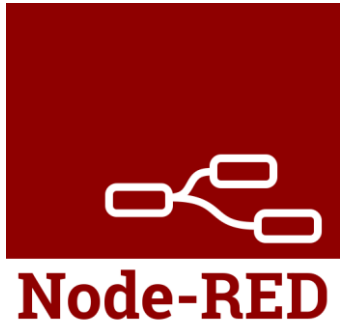
New developers & education

- Short learning curve
- Easy to use
- Low barrier to entry

App Developers

- Rapid prototyping
- Easy to integrate with existing tools and applications
- Easy to extend with richer/bespoke functionality

Node-RED



Community developers

- Open standards
- Flexibility
- Ability to share

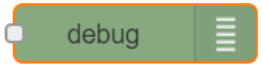
Hackers

- Runs on Raspberry Pi, Beaglebone, C.H.I.P., other low power devices
- Works with Arduino, etc…

Basic Node types

Inject node

- Allows manual triggering of flows
- Can inject events at scheduled intervals



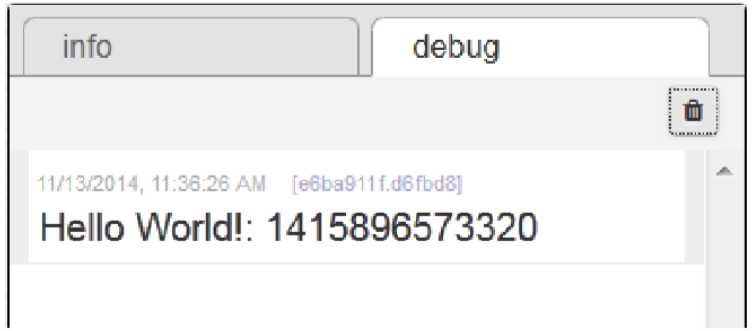
Debug node

- Show message content; either payload or entire object

Template Node

- Modifies the output based on a Mustache Template

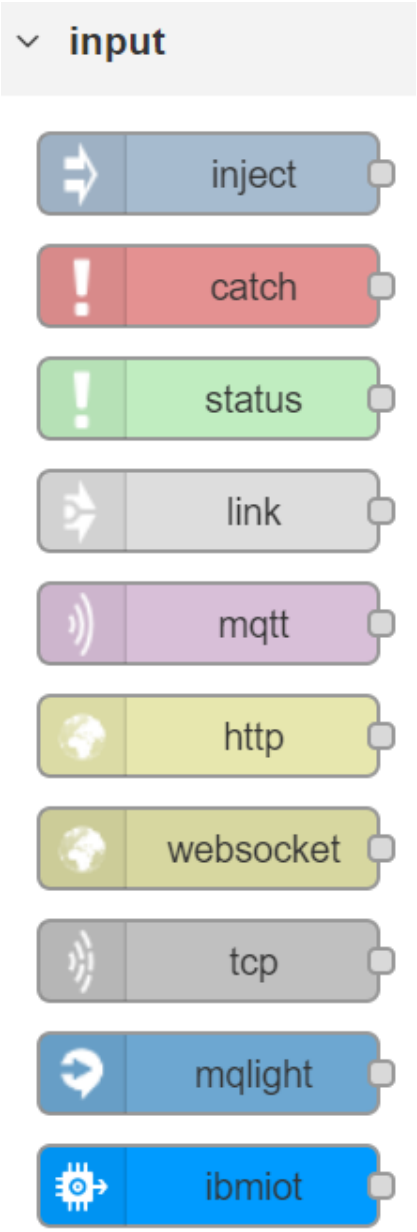
Node-RED Hello World



When you click on the Inject Node, it sends and event through the flow – triggering the template node and sending the result to the Debug node

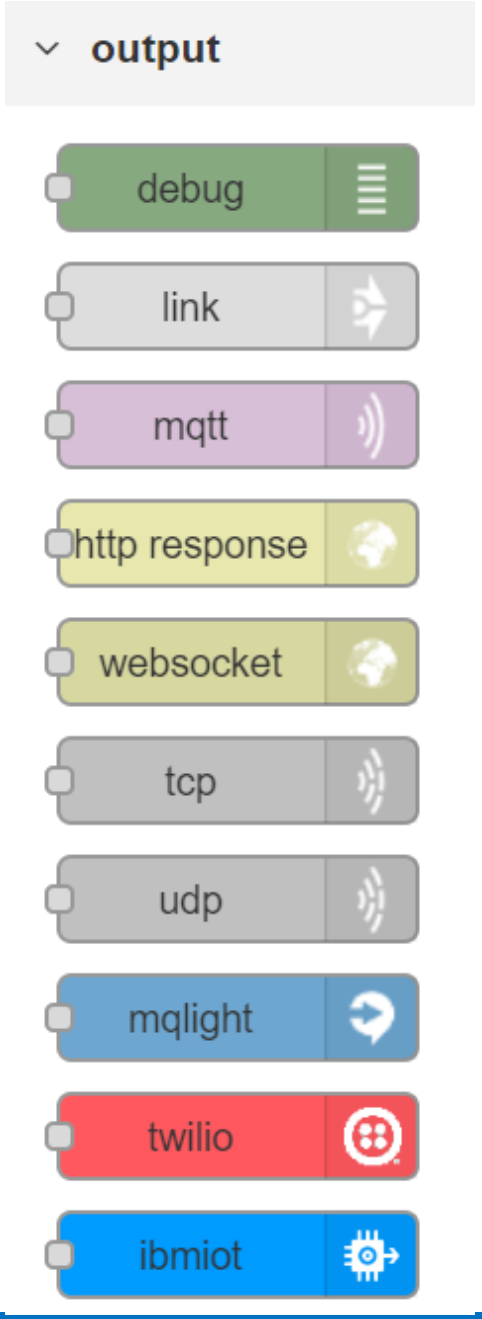
Other input nodes

- HTTP – Act as an HTTP endpoint; great for building RESTful services
- IBMIOT – Receive messages from an attached IOT Foundation account
- Also can receive from Websockets, MQTT (pick your own broker), TCP and MQ Light



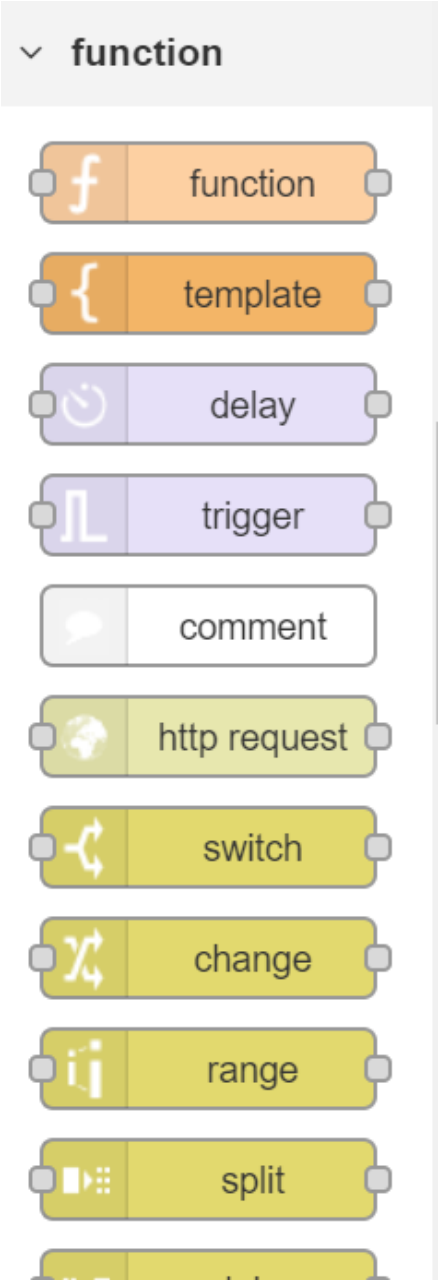
Other output nodes

- HTTP Response; required as the final node when the input comes from an HTTP Request
- IBMIOT – send events out to the attached IOT Foundation account
- Twilio – send SMS messages via the Twilio service
- IBM Push – Send Push notifications to mobile devices
- Also can send requests through TCP, UDP, MQLight, WebSockets.



Function node types

- Function node
- Run user-defined node.js code on the messages going by
- Uses vm.createScript under the covers to sandbox execution
- Console, util, Buffer included for convenience
- Switch node changesflow to different options based on a comparison

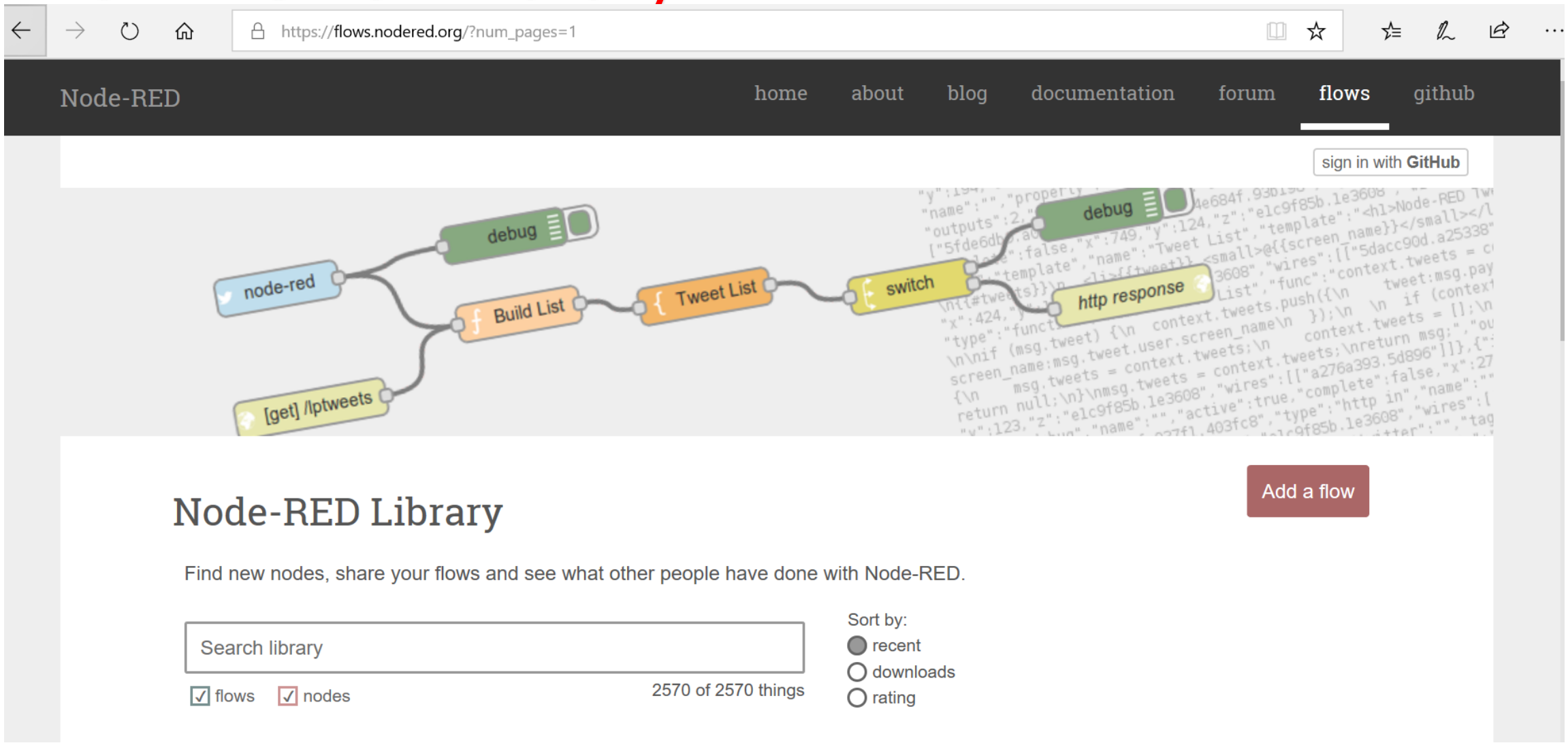


Creating your own nodes

https://nodered.org/docs/creating-nodes/first-node.html

- Easy to wrap any npm module into a palette node
- Each node is defined in a pair of files
- .js: server-side behavior
- .html: appearance in editor and help
- Can be shared and installed via npm
- npm install node-red-node-xmpp

Online flow library

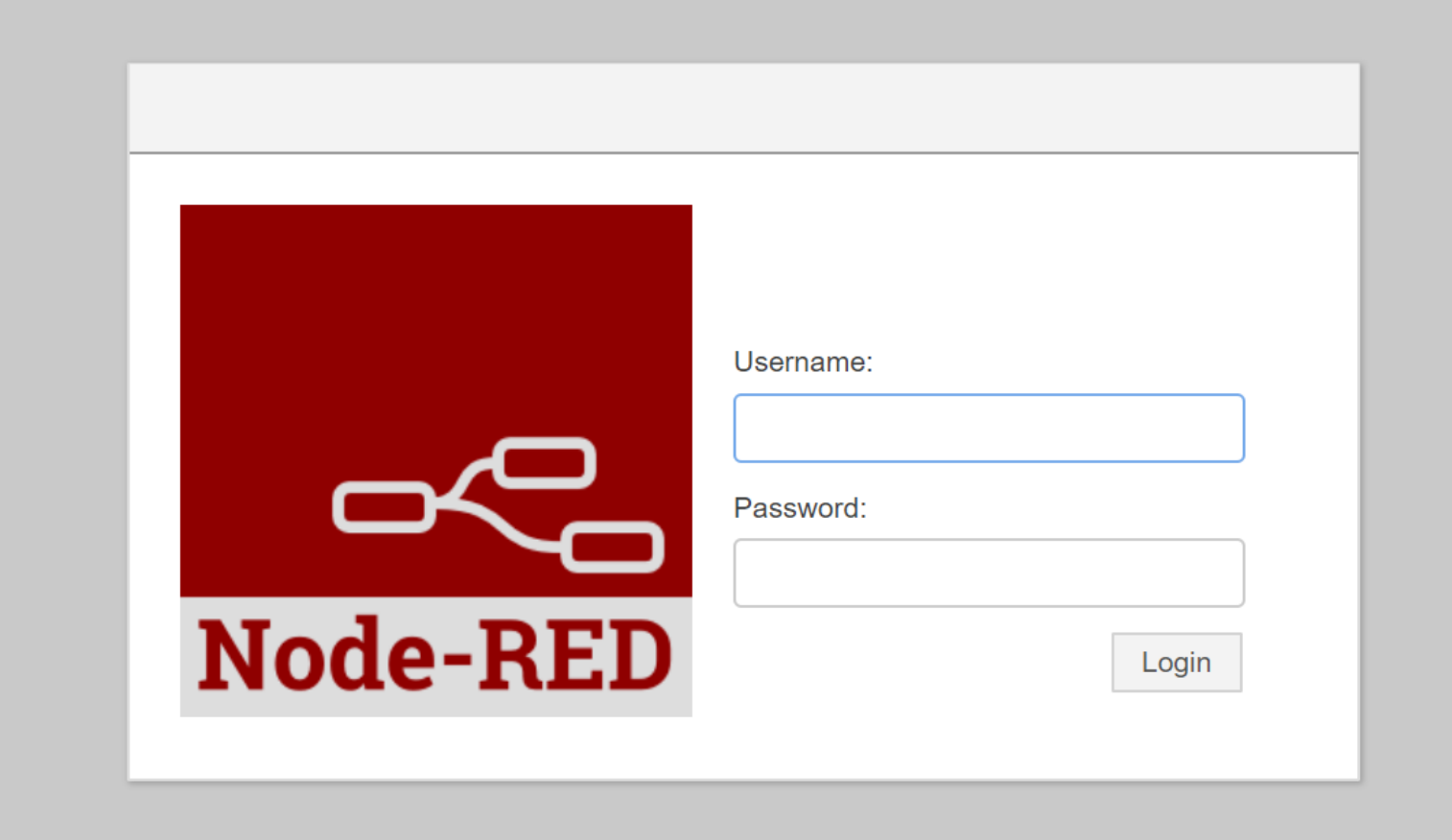


Contributors add flows through Github

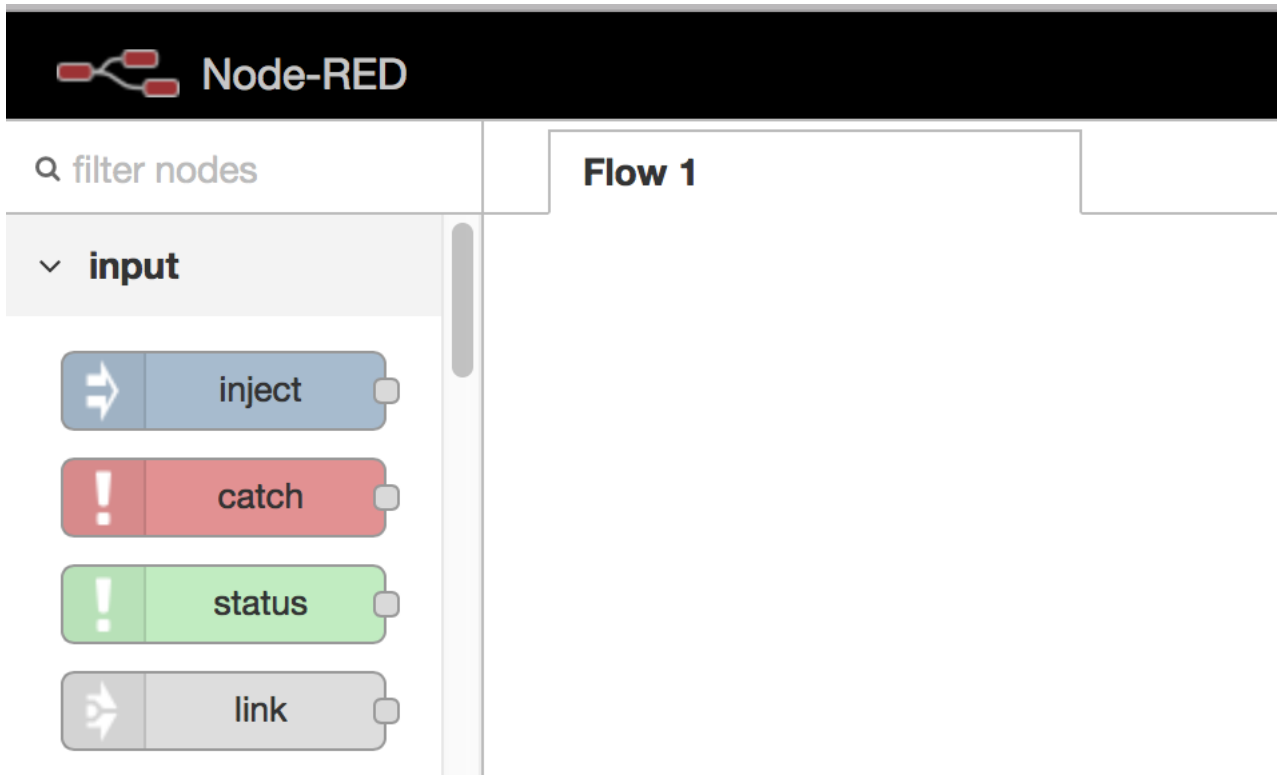
FIRST FLOW

Create your first flow

- Logon to the editor

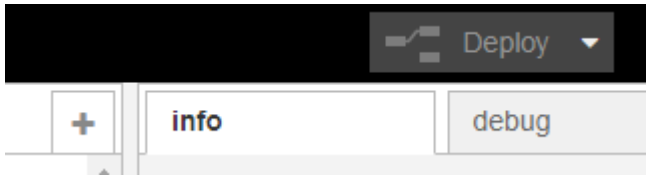


Create your first flow



Create your first flow

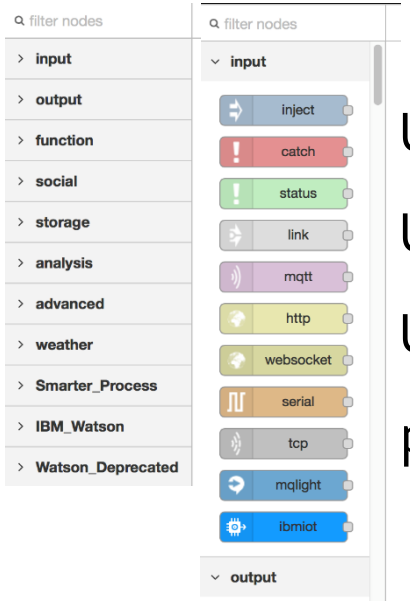
- Click the **Add** icon (+) to create a new flow.



- An application in Node-RED is called a *flow*.

Create your first flow

- The palette in the left column shows you all the available nodes.
- The nodes are grouped by category. The main categories of nodes are input, output, and function.



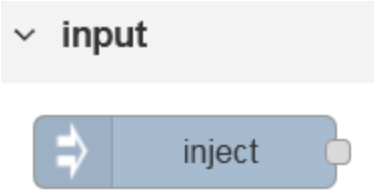
Use input nodes to input data into a Node-RED application, or flow.

Use output nodes to send data outside of a Node-RED flow.

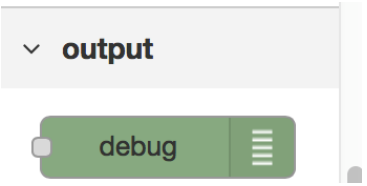
Use function nodes to process data. You can use the function node to pass messages though a JavaScript function.

Create your first flow

- Select an input **inject** node and drag it onto the canvas.



- Select an output **debug** node and drag it onto the canvas.

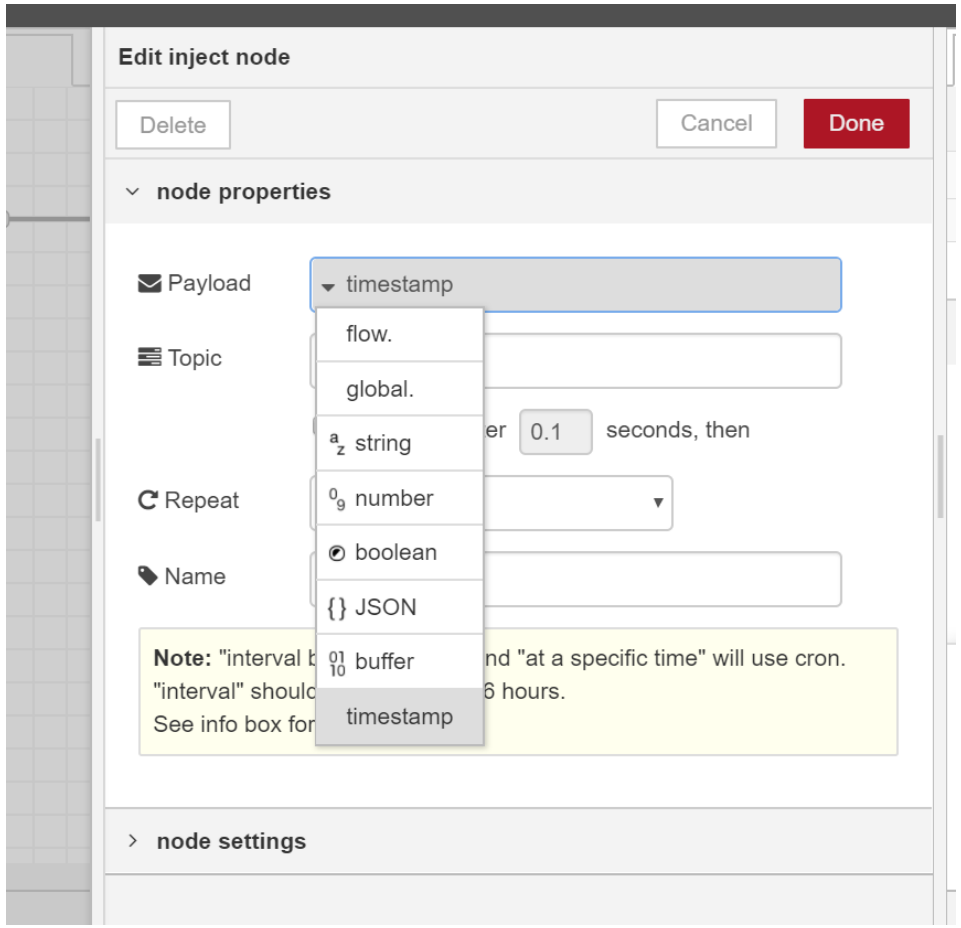


Create your first flow

- Link, or wire, the two nodes together by clicking and dragging your cursor from one node to the other. Note that the debug and inject nodes change their display names when you drag them onto the canvas. This name change is expected and shows additional context for the node.

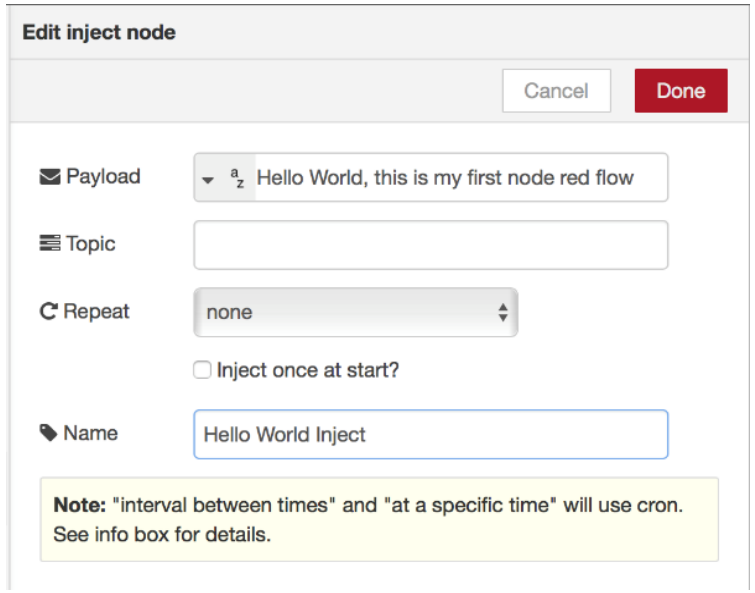
Create your first flow

- Double-click the **timestamp** node. For the **Payload** field, select **string**.

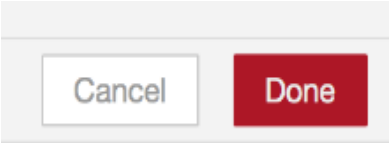


Create your first flow

- Enter a string, such as Hello, this is my first Node-RED application.
- In the **Name** field, enter a name for this node, such as Hello World inject.



- Click **Done**.



Create your first flow

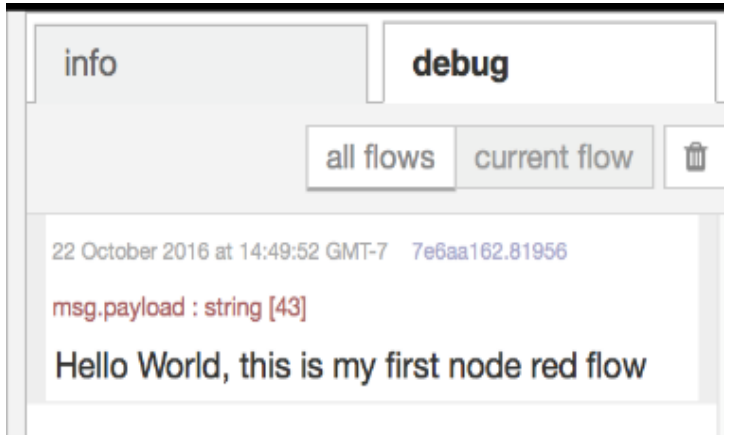
- The blue circles indicate that your flow has unsaved changes, which means that the application needs to be deployed.

- Click **Deploy** to deploy and save your changes.

Create your first flow

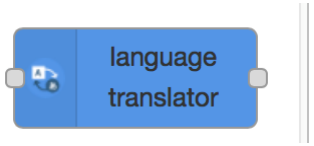
- The **debug** node writes to the **debug** tab, which helps you monitor the flow through your application.
- To initiate the flow, click the tab linked to the **inject** node.+

- You now see the output on the debug tab.



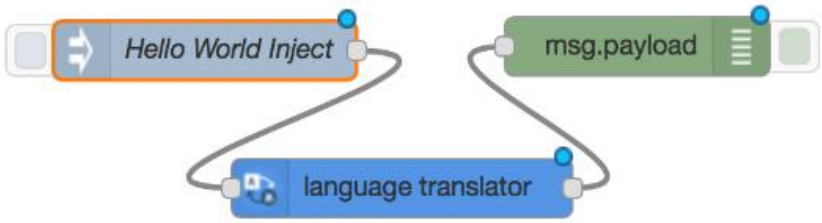
Create your first flow

- In the **filter nodes** search field, enter translator to find the **language translator** node.



Drag the node onto the canvas so that it lies in between the **inject** and **debug** nodes. You can move the nodes to make more space.

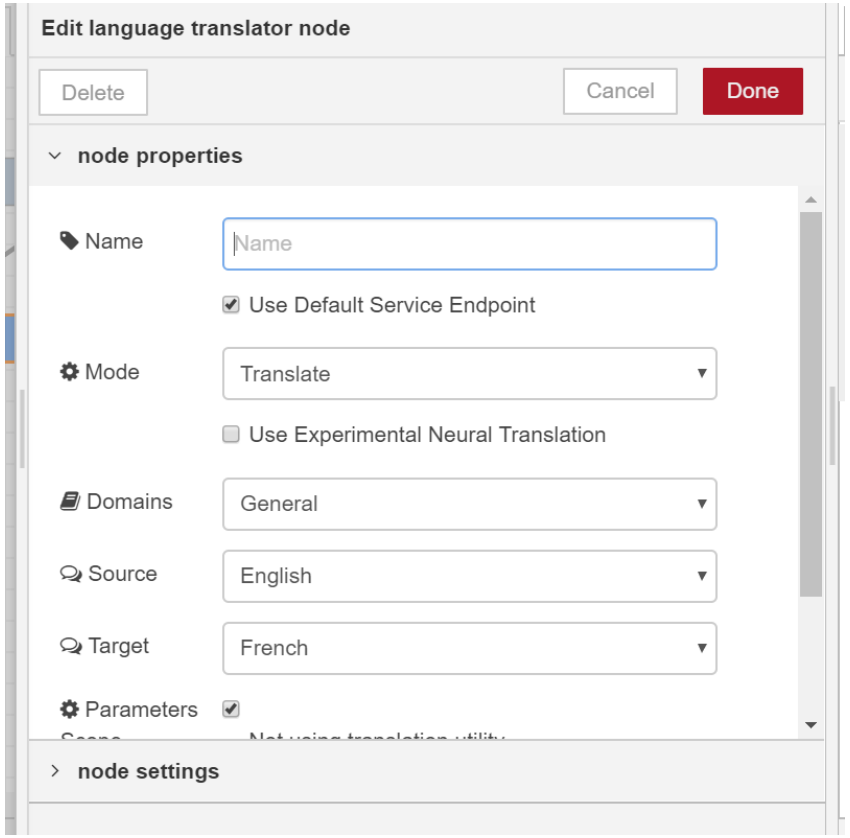
To remove an unwanted line, select the line and press Delete on your keyboard.



Create your first flow

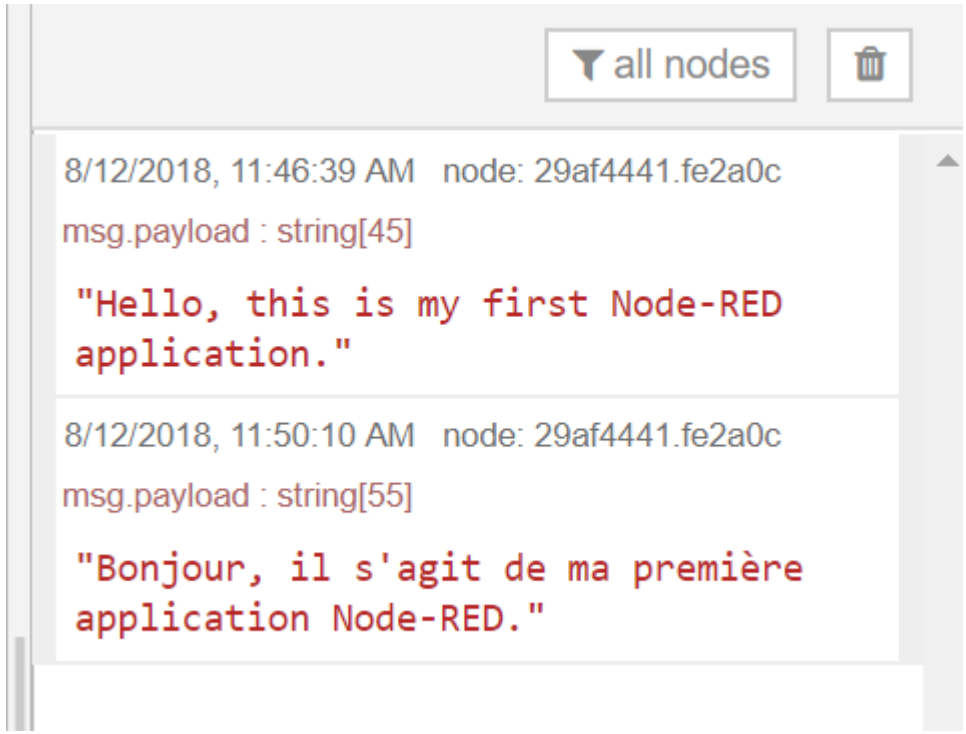
Double-click the **language translator** node. Select to translate from the inject English to another language.

Click **Done** to save your changes



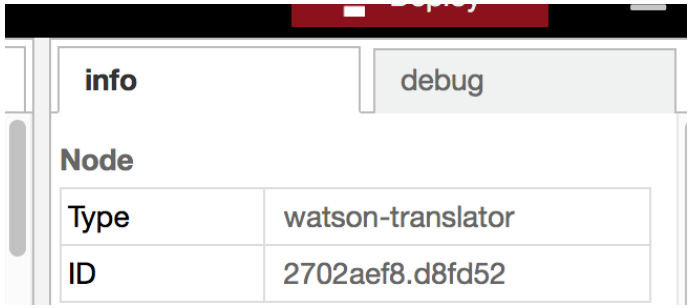
Create your first flow

Deploy and click on the input node gives us :



Create your first flow

- Select the **language translator** node and click the **info** tab.



Notice that the node puts its translated output in msg.translation. **msg** is a reserved object that Node-RED uses to allow individual nodes to communicate with each other. Think of **msg** as an envelope into which one node places information that allows another node to read it. The **language translator** node is expecting to find a payload that is already in the msg envelope, and it will insert a translation into the msg envelope.

Create your first flow

The Watson Language Translator service enables you to translate text from one language to another and to add your own translation models.

Translation Mode.

The text to translate should be passed in on `msg.payload`.

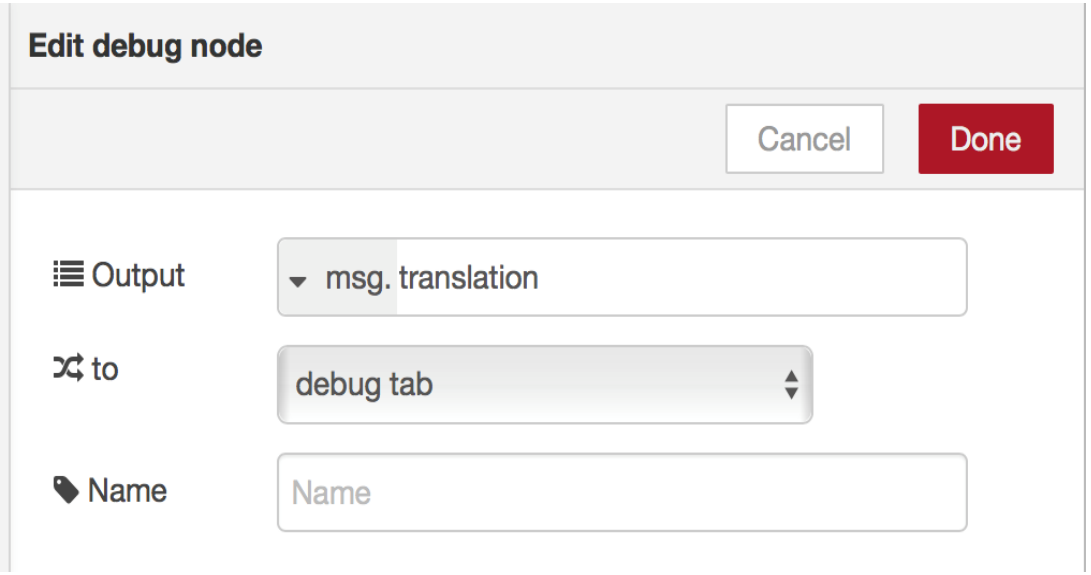
The translated text will be returned on `msg.payload`.

The full response from the service will be returned on `msg.translation`

Source and destination language parameters

Create your first flow

- Open the **debug** node and change the output to msg.translation. Enter the word translation after msg. Click **Done** to save your changes. Then, deploy your flow.



Create your first flow

Initiate the flow by clicking the tab on the **inject** node.

- View the translated text in the **debug** tab. The application is translating the text that you entered in the **Payload** field of the **inject** node.

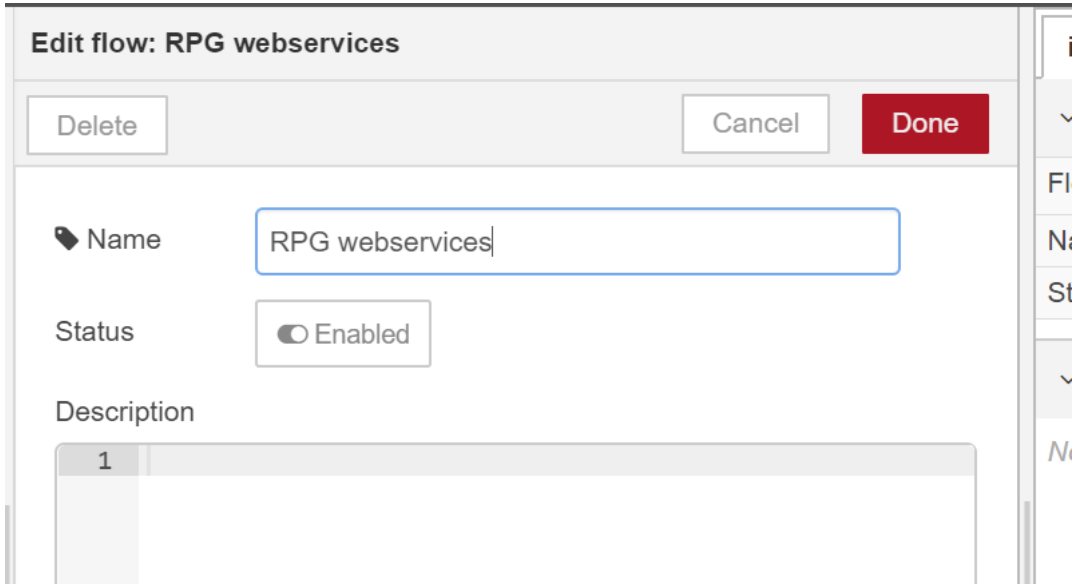

```
8/12/2018, 12:01:23 PM   node: 29af4441.fe2a0c
msg.translation : Object
▾object
▾response: object
  ▾translations: array[1]
    ▾0: object
        translation: "Bonjour, il
        s'agit de ma première
        application Node-RED."
    word_count: 7
    character_count: 45
```

RPG WEBSERVICES

RPG webservices

- Create a new flow tab by clicking +.

- Double-click the new tab and enter a name for the new flow tab. Then click **Done. If the edit screen does not appear click the right menu and rename.**

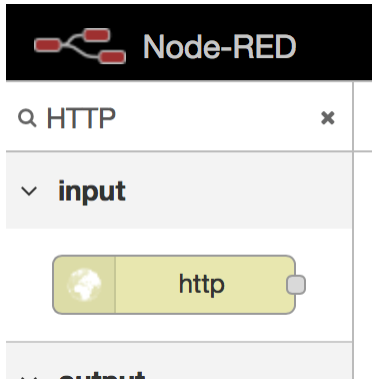


RPG webservices

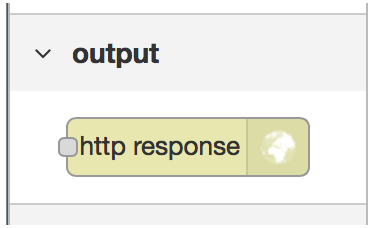
- Drag and drop a comment node onto the canvas and change the title of the node to **Display an initial web page**

RPG webservices

- Drag and drop an input **http** node onto the canvas. Use the **filter nodes** search field to find the nodes.

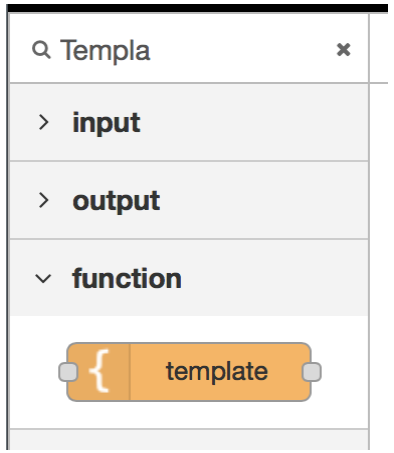


- Drag and drop an output **http response** node onto the canvas.

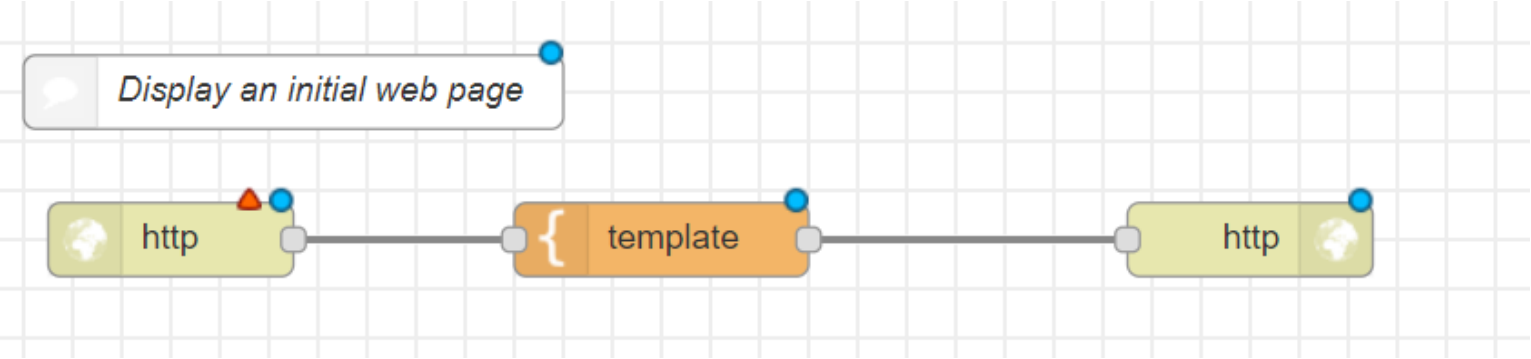


RPG webservices

- Drag and drop a **template** node onto the canvas between the **http** and **http response** nodes.

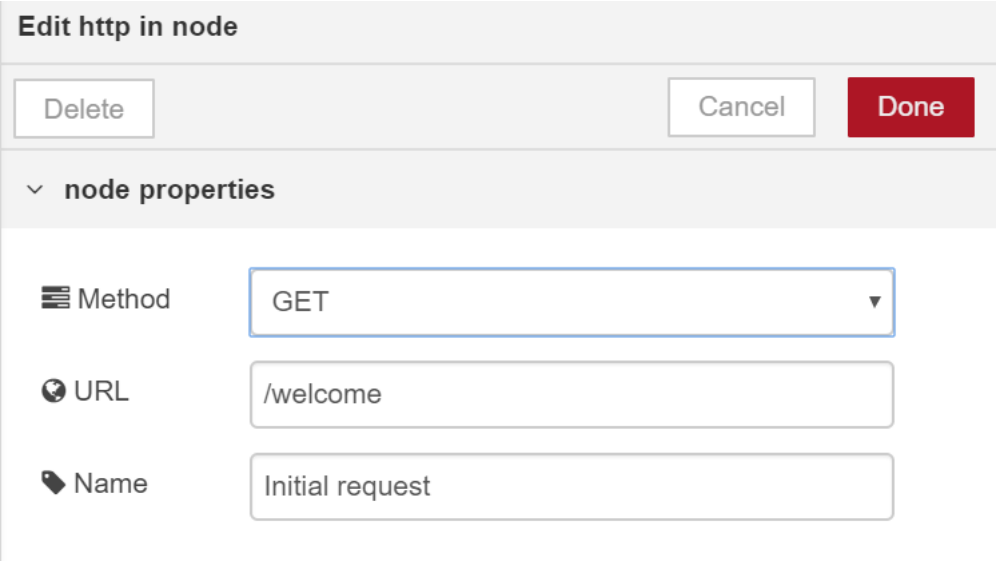


- Wire the three nodes together.



RPG webservices

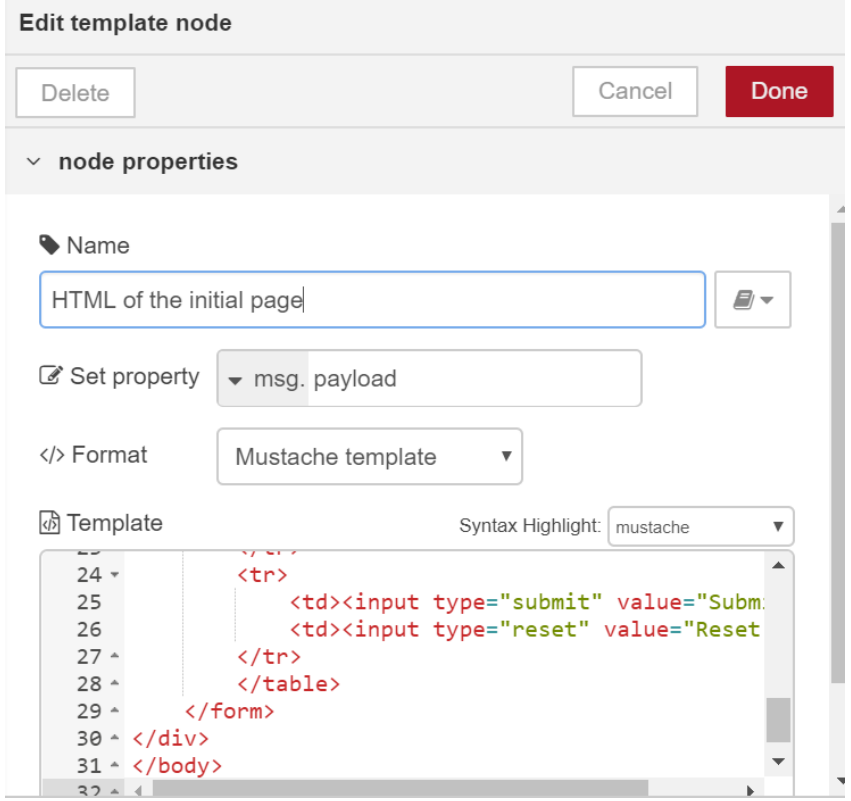
- Double-click the input **http** node. Edit it to create an HTTP route to your web page by entering /welcome in the **URL** field. Enter **Inital request** as name.



- Click **Done**

RPG webservices

- Double-click the **template** node to edit it. Copy the HTML code from the intial html file and click done.

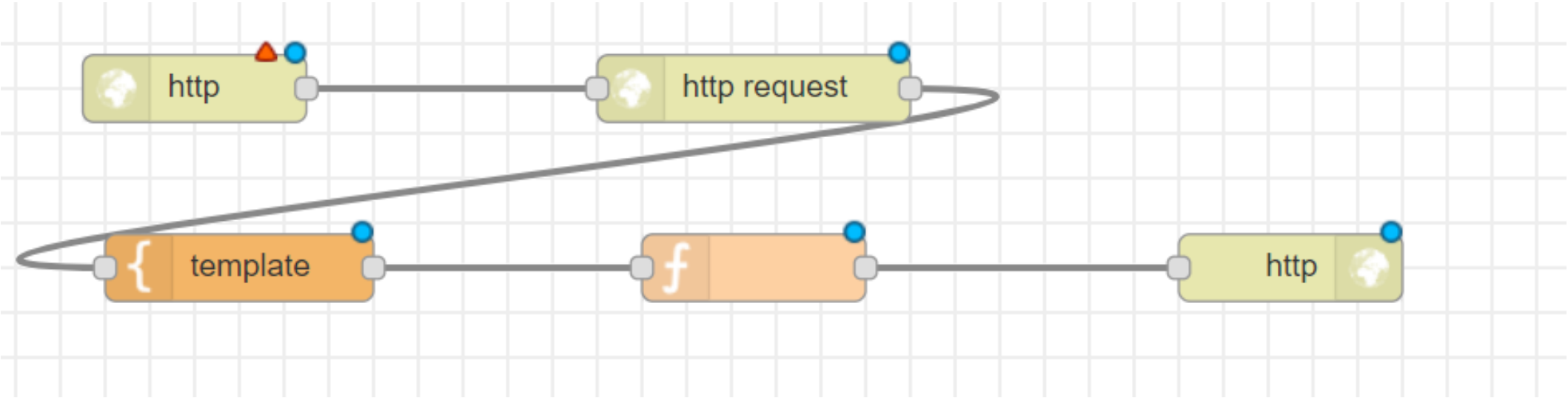


RPG webservices

- Drag and drop a comment node onto the canvas and change the title of the node to **Query the customer information by using received ID as the key**

RPG webservices

- Drag a HTTP in, HTTP request, template, function and HTTP ouput node on the canvas and tie them together.



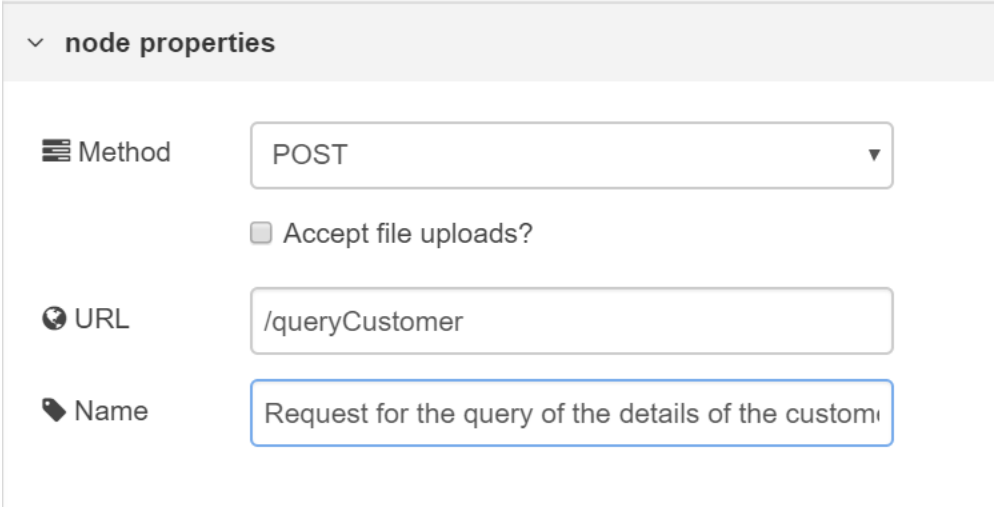
RPG webservices

- Change the properties of the HTTP input node into

- method = post

- url /queryCustomer

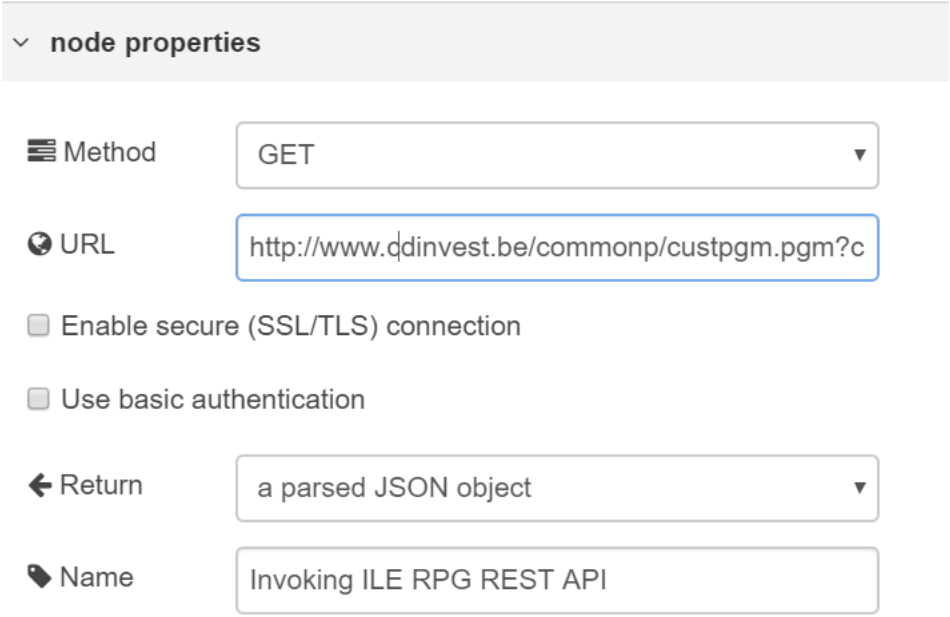
- name Request for the query of the details of the customer



RPG webservices

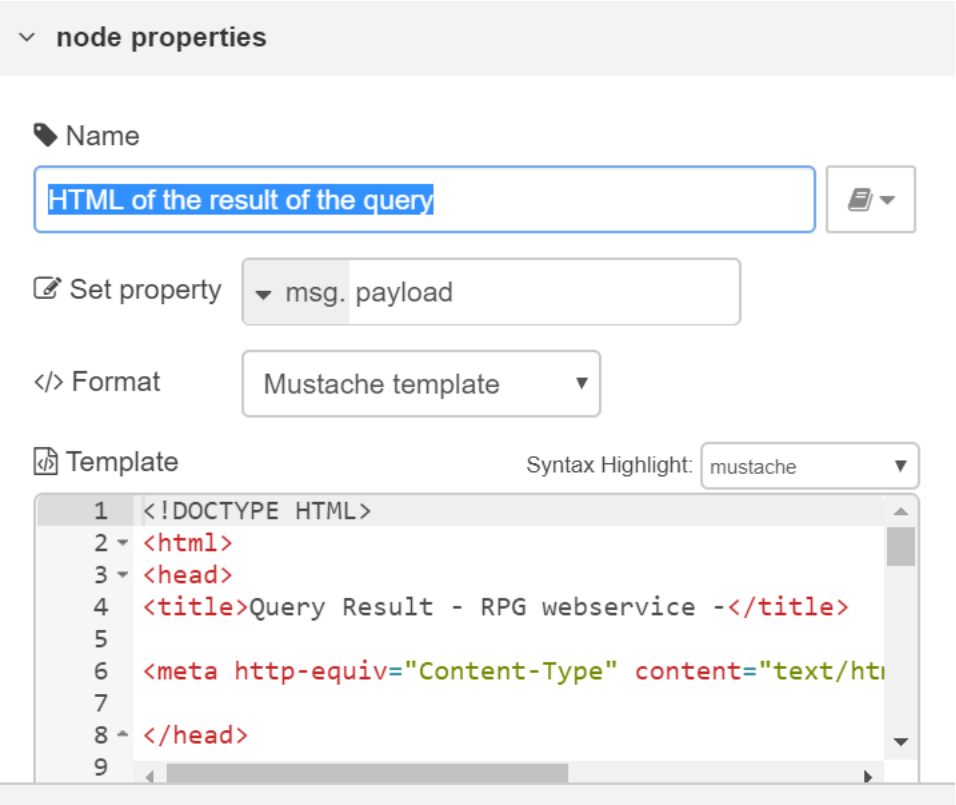
- Edit the HTTP request node as follows :

- Method = get
- URL = http://www.cdinvest.be/commonp/custpgm.pgm?customerid={{payload.customerID}}
- Return = a parsed JSON object
- Name = Invoking ILE RPG REST API



RPG webservices

- Set the name of the template node to HTML of the result of the query and copy the output.html file content into the template field.



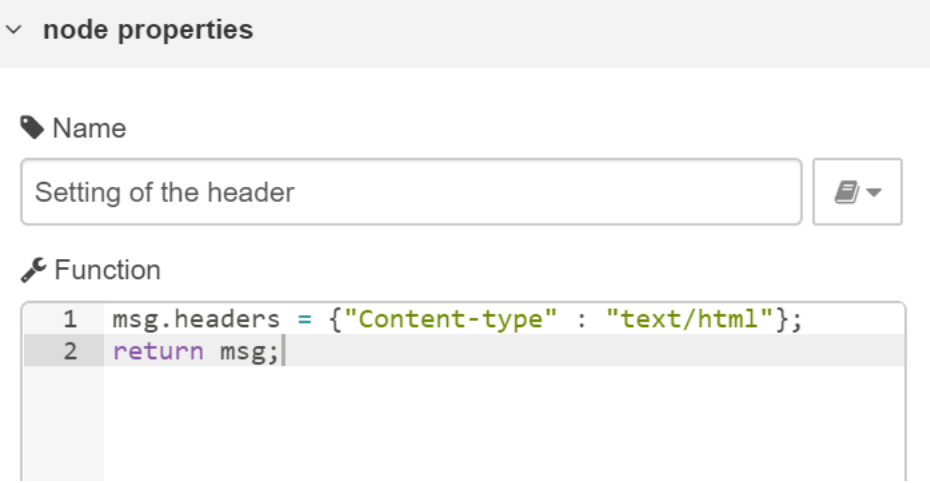
RPG webservices

- Set the name of the function node to “setting of the header” and copy the function :

msg.headers = {"Content-type" : "text/html"};

return msg;

Web browsers need a valid content header.



RPG webservices

- Deploy your changes.
- Test your application by entering 938472 as customerid.

welcome - RPG webservice call -

Enter Customer ID you want to see the details.

Customer ID (*) 938472

Submit Reset Input

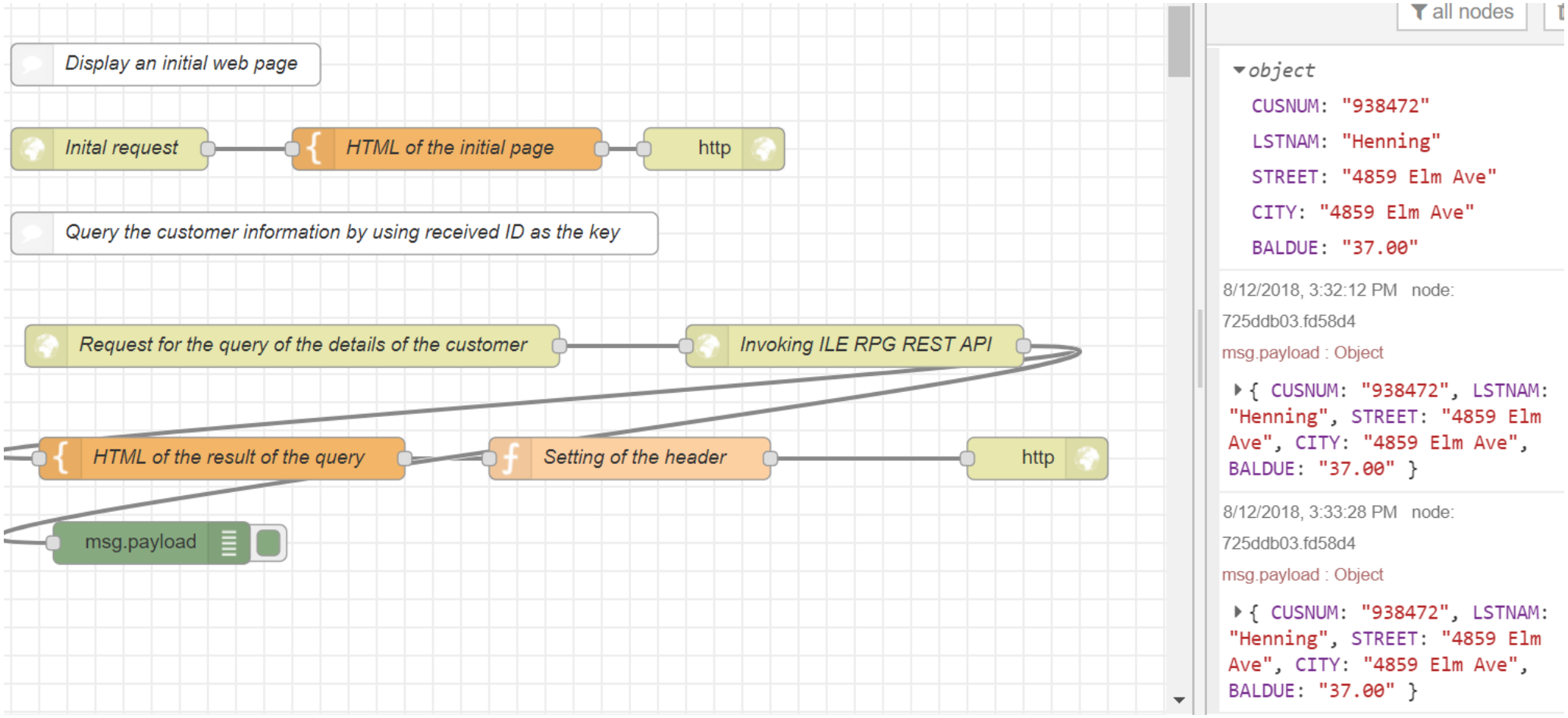
Query Result - RPG webservice -

Customer ID: 938472
The details of the Customer

Customer ID	938472
Customer Name	Henning
Customer Street	4859 Elm Ave
Customer City	Dallas
Balance due	37.00

RPG webservices

- Add a debug node, link it to the request output, you can then see debug output as well



RPG webservices

- Add a template node to display an error message in case the customer was not found and insert the notfound html

Name

HTML customer not found

Set property msg. payload

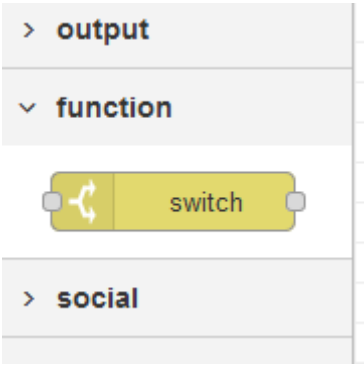
Format Mustache template

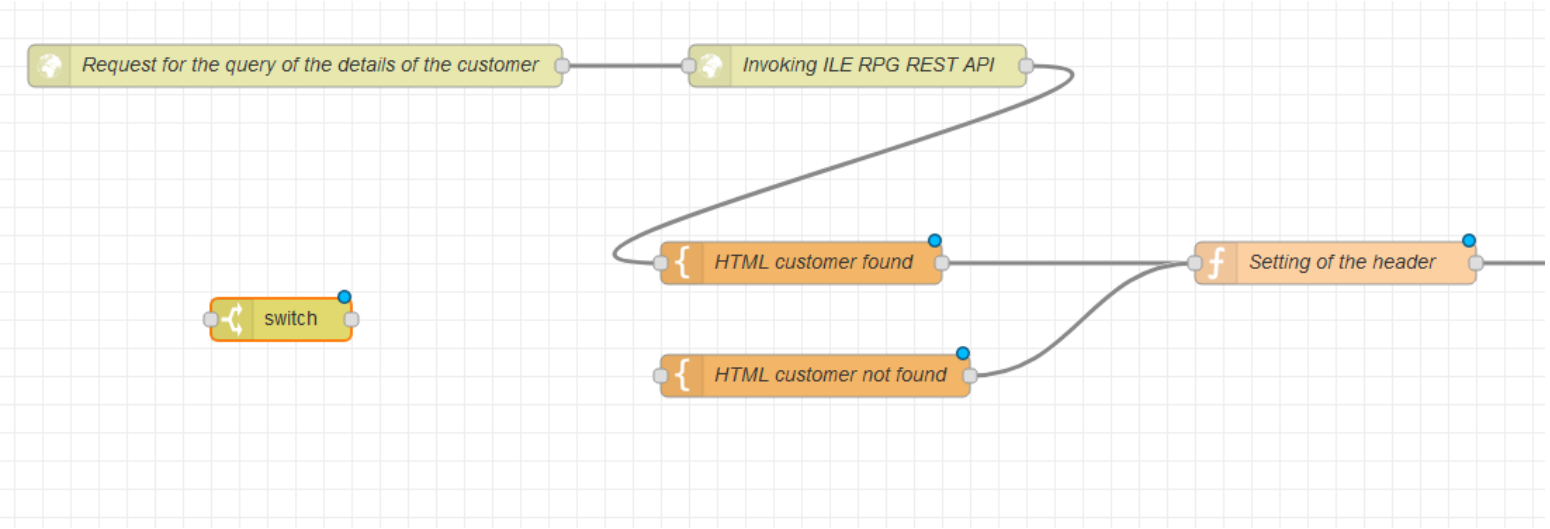
Template Syntax Highlight: mustache

```
<!DOCTYPE HTML>
<html>
<head>
<title>Query Result - RPG webservice -</title>

<meta http-equiv="Content-Type" content="text/html;

</head>
<body>
<div style="text-align:center">
    <hr/>
    <h2>Query Result - RPG webservice -</h2>
    <hr/>
    Customer not found !
</div>
</body>
</html>
```

RPG webservices

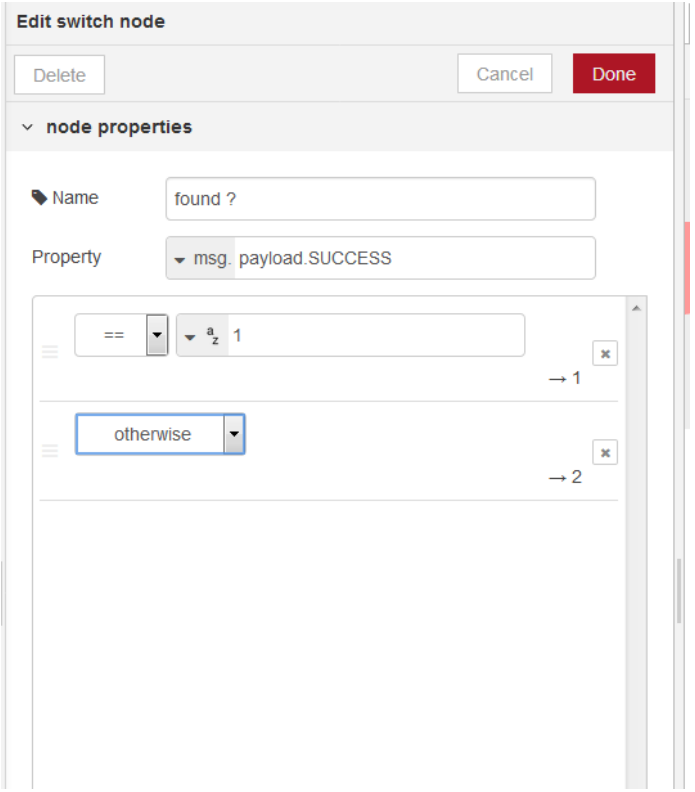
- Remove the debug node and link, remove the link from the http request and add a switch node.





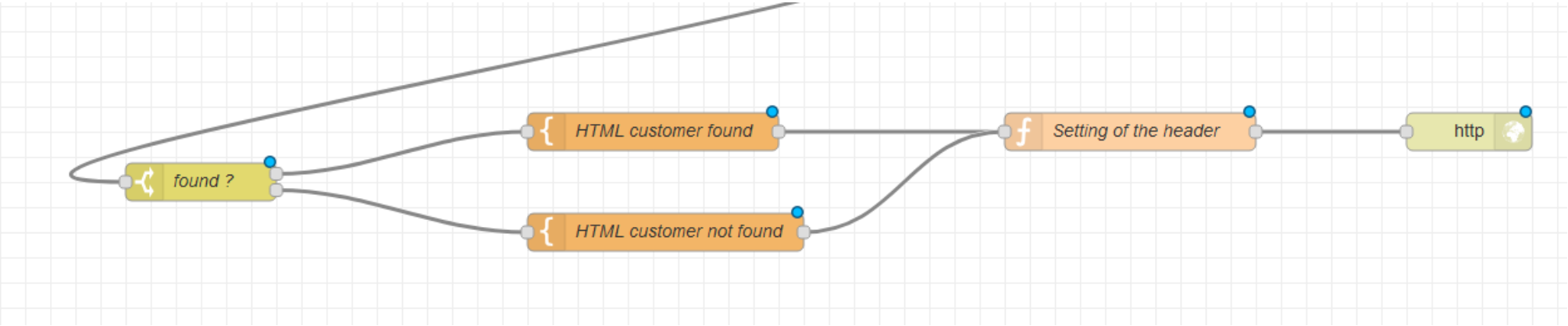
RPG webservices

- Link the request node to the input of the switch node. Edit the switch node as follows and click done.



RPG webservices

- Link the the top output of the switch node to the original html ouput, link the bottom node to the new not found template. Link the new not found template to the http header function and click Deploy.



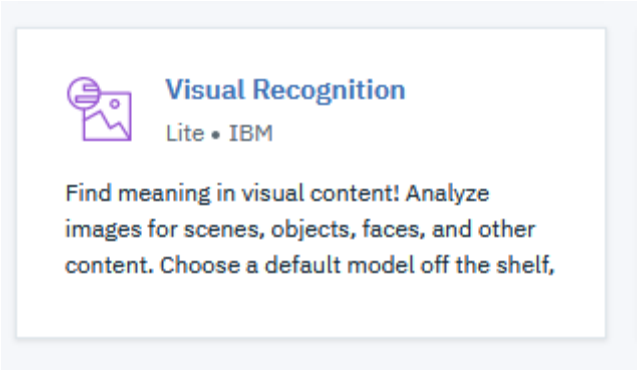
RPG webservices

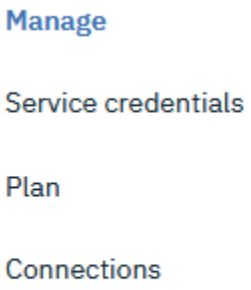
- Deploy your changes.
- Test your application by entering 123456 as customerid.

VISUAL RECOGNITION

Visual recognition

- Go to the IBM Cloud catalog and search for the Visual recognition service.

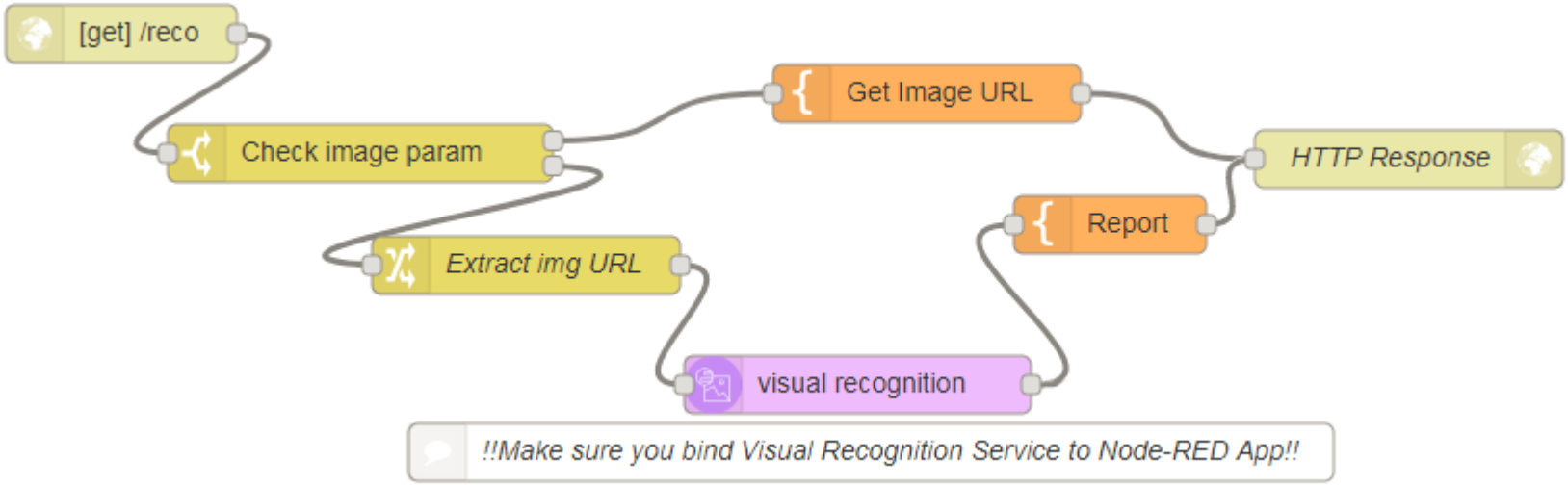




- Click **connections** after the create. Select your instance and restage the node-red instance. On your own server you will have to enter the credentials on the visual recognition node.

Visual recognition

- The flow will present a simple web page with a text field of where to input the image's URL, then submit it to Watson Visual Recognition. It will output the labels that have been found on the reply Web page.

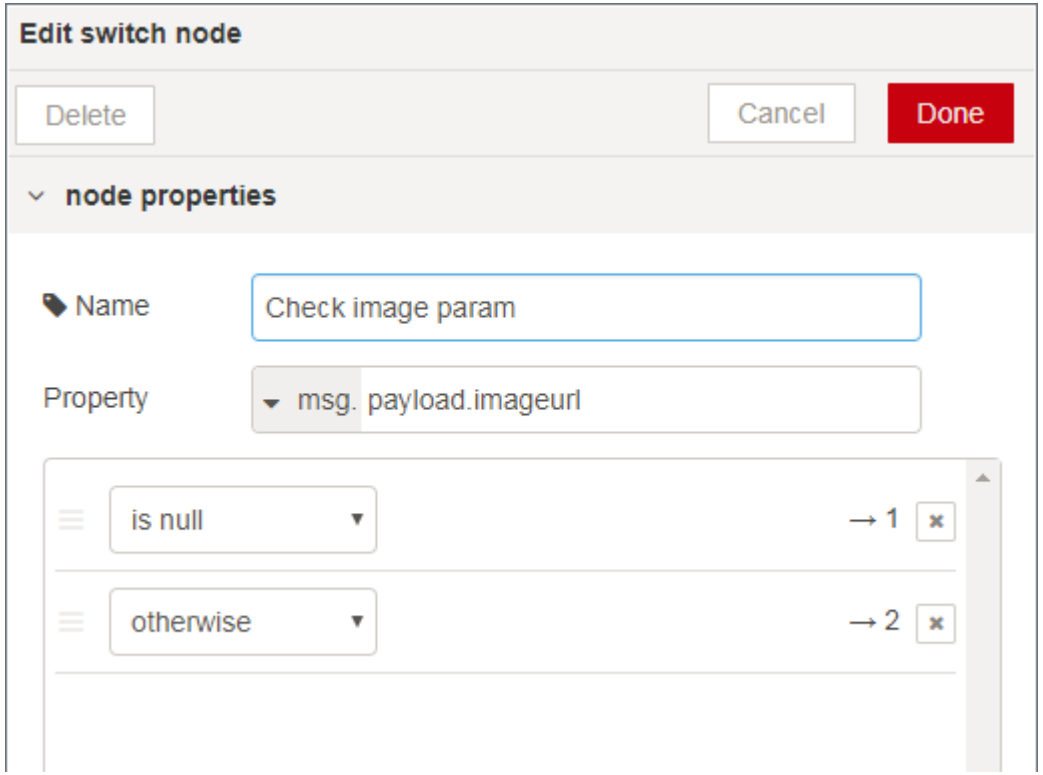


Visual recognition

- The nodes required to build this flow are:

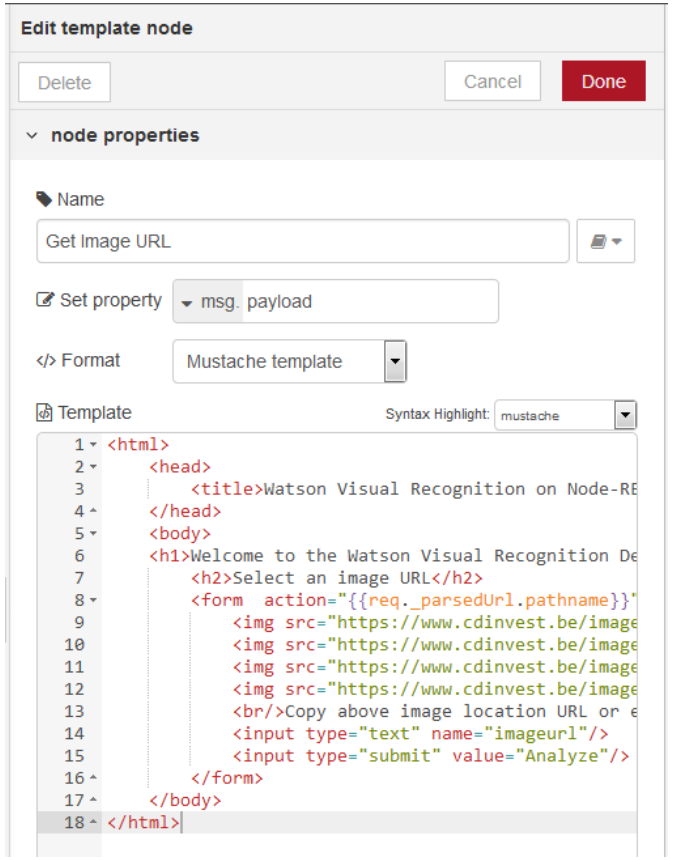
node, configured with a /recognition URL

node to test for the presence of the imageurl query parameter



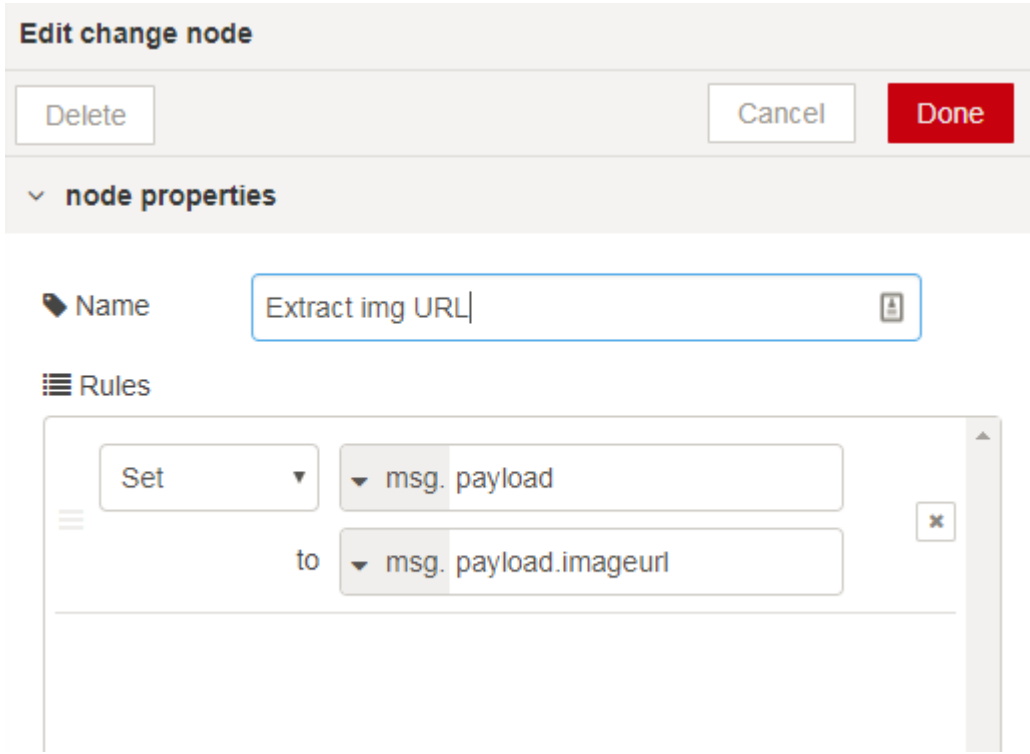
Visual recognition

node, configured to output an HTML input field and suggest a few selected images stored on our webserver. (visual1.html) Linked to the top output of the switch node. Change the output as to blanks or plain text !

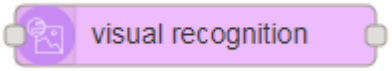


Visual recognition

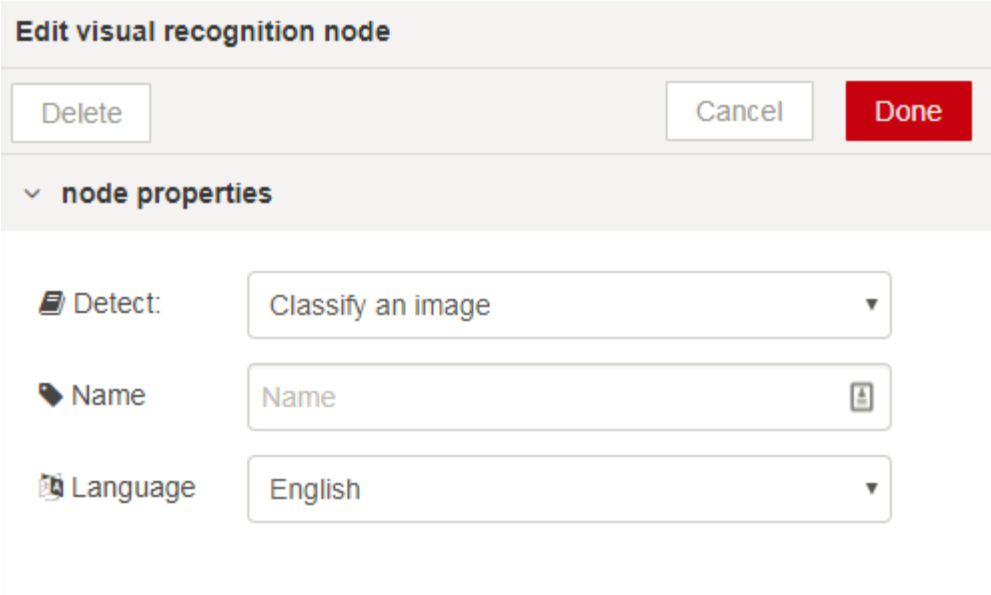
node (named Extract img url here) to extract the imageurl query parameter from the web request and assign it to the payload to be provided as input to the visual recognition service.



Visual recognition

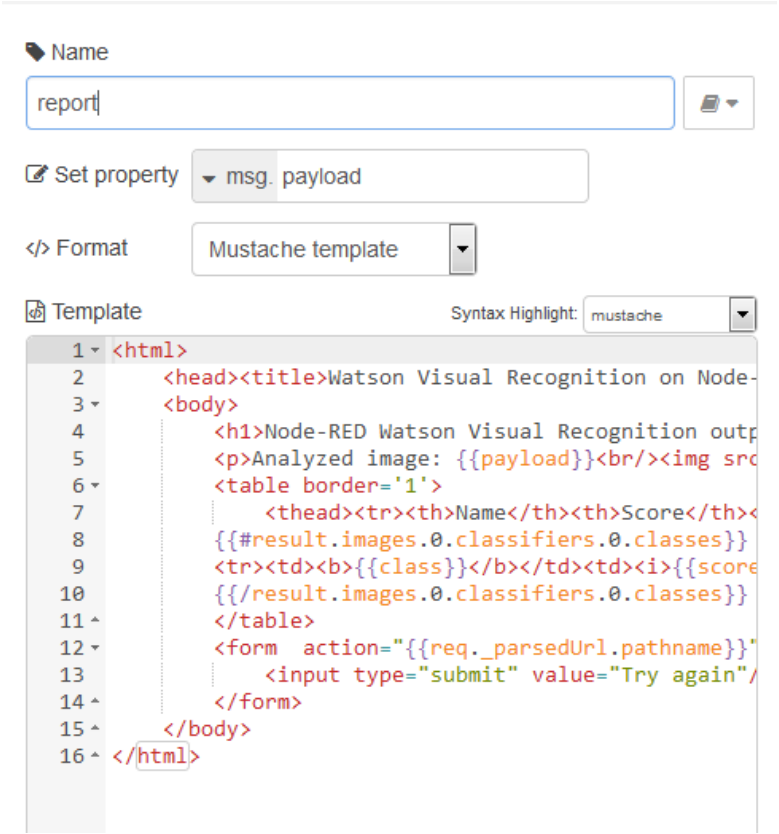


node. Make sure that the credentials are setup from IBM Cloud, i.e. that the service is bound to the application. This can be verified by checking that the properties for the Visual Recognition node are clear:



Visual recognition

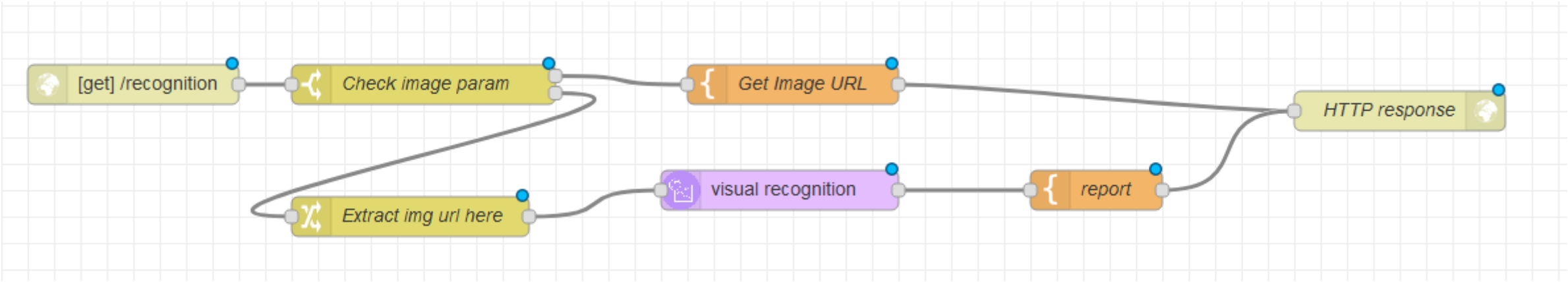
node, configured to output an HTML which returns the output returned from the visual recognition node. (report.html)



Visual recognition

- 

de, linked with both template nodes to output the HTML.



Visual recognition

- Click Deploy
- To run the web page, point your browser to https://XXXXX.eu-gb.mybluemix.net/recognition and enter the URL of some image.
- The URL of the pre-selected images can be copied to clipboard and pasted into the text field.
- The Watson Visual Recognition API will return an array with the recognized features, which will be formatted in a HTML table by the template. A print screen can be found on the next slide.

Visual recognition

Node-RED Watson Visual Recognition output

Analyzed image: https://raw.githubusercontent.com/watson-developer-cloud/visual-recognition-nodejs/master/public/images/samples/2.jpg

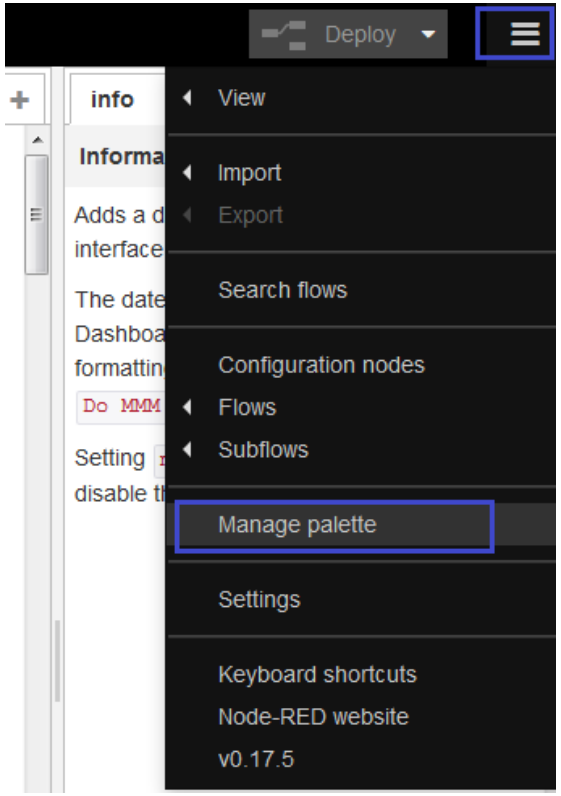
Name	Score
clothing store	0.948
shop	0.951
retail store	0.973
building	0.973
department store	0.533
dressing room	0.5
indoors	0.5
sage green color	0.838
charcoal color	0.607

Try again

DASHBOARD

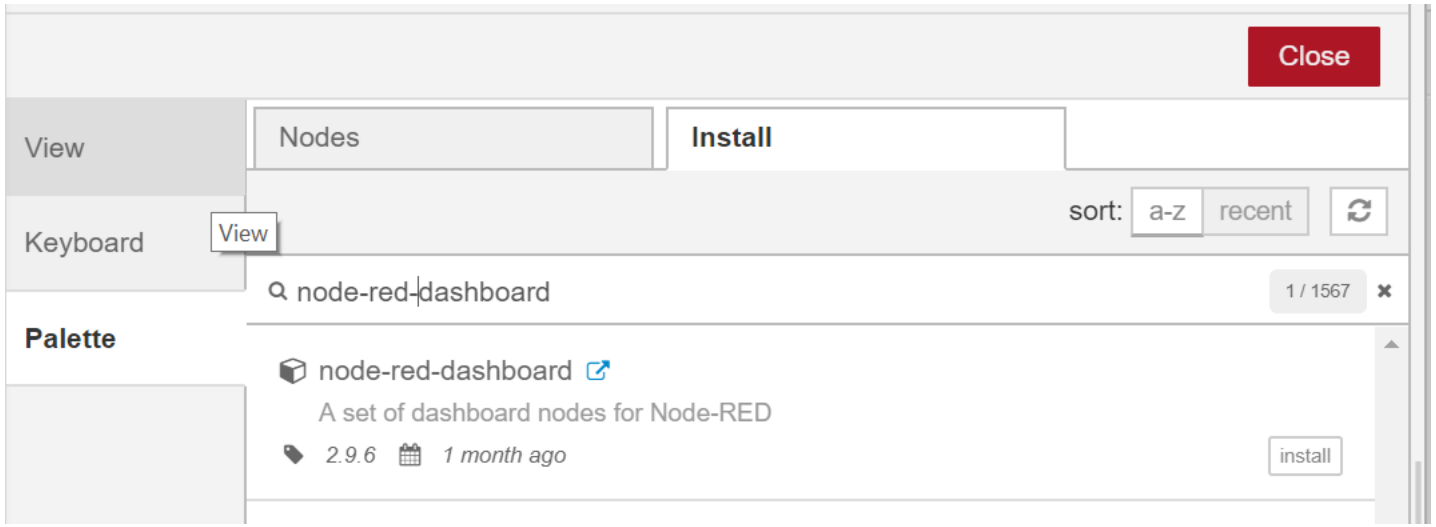
Dashboard

First, we need to install the *node-red-dashboard* node in our Node-RED palette. In Node-RED, click the button at the upper-right corner and click **Manage palette**.



Dashboard

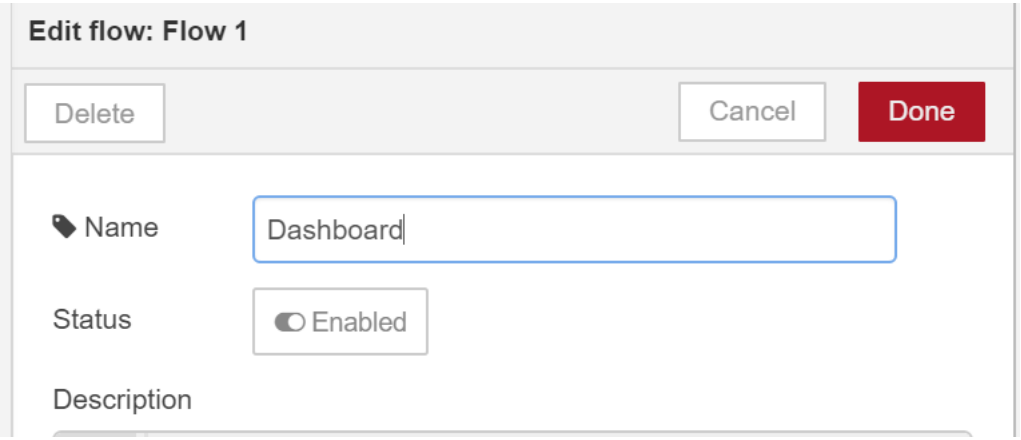
- On the **Install** tab, search for *node-red-dashboard*, and click **Install**. Wait for the installation confirmation dialog box to be displayed, and then restart Node-RED if necessary.



Dashboard

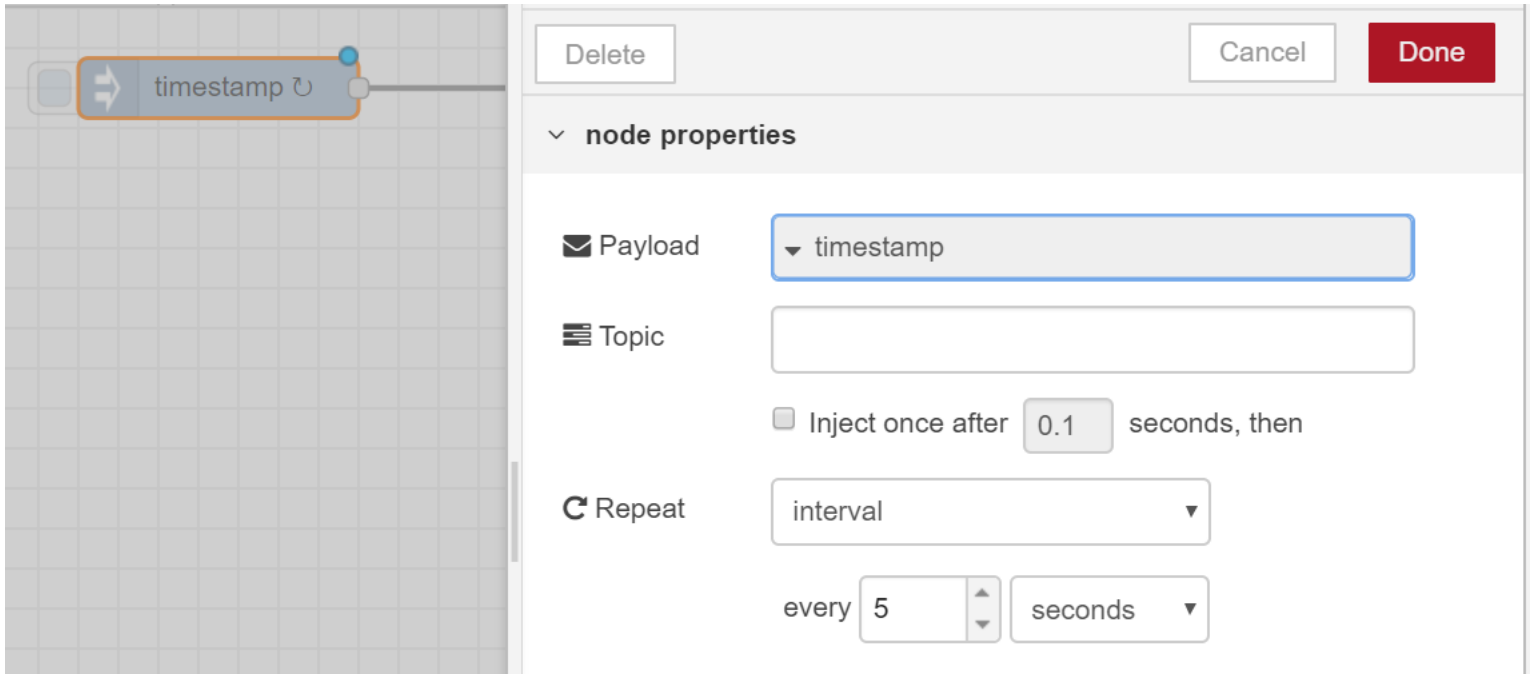
- Create a new flow tab by clicking +.

- Double-click the new tab and enter a name for the new flow tab. Then click **Done. If the edit screen does not appear click the right menu and rename.**



Dashboard

- Set up a simple flow to send a random number every 5 seconds to a chart.
- Add an inject node to send a timestamp every 5 seconds by setting the ***payload*** to timestamp and the ***repeat*** field to an interval of 5 seconds.



Dashboard

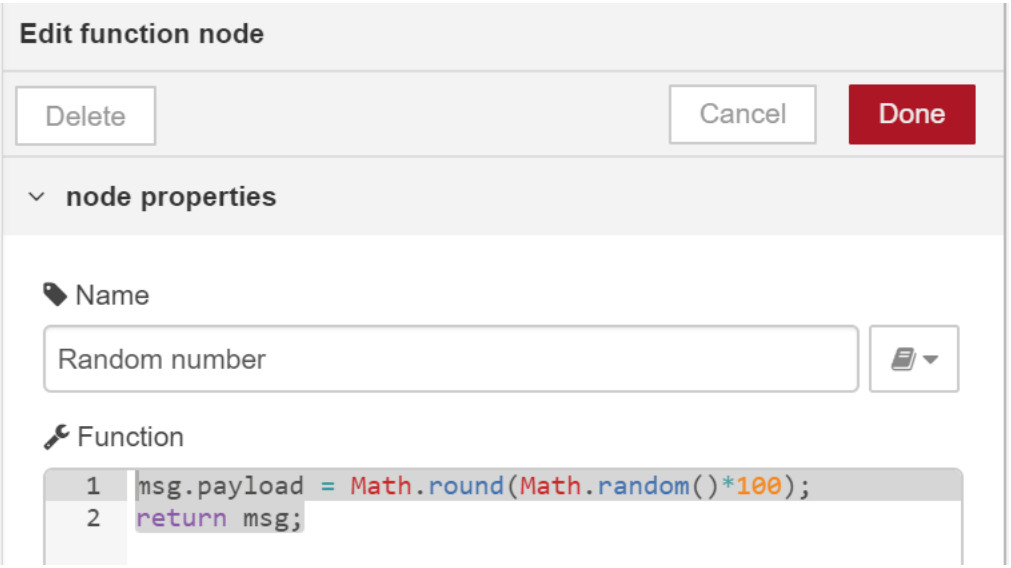
- Add a function node 

to return a random number. Link it to the inject node

Function :

msg.payload = Math.round(Math.random()*100);

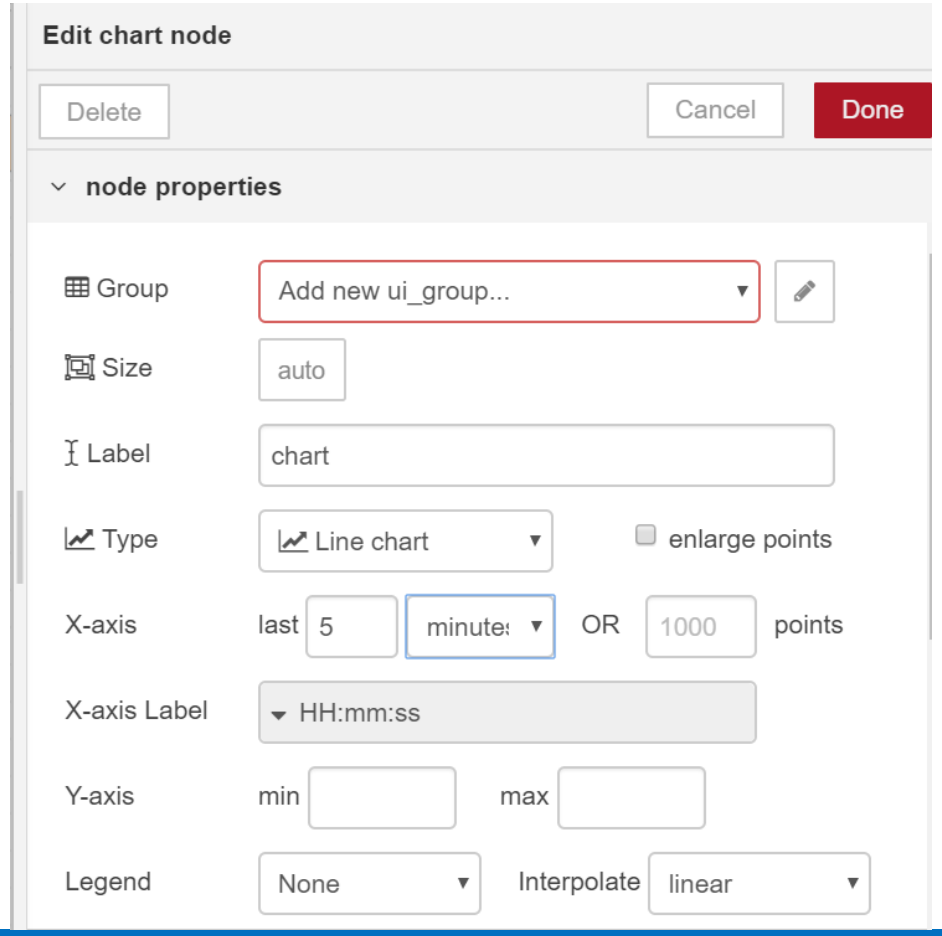
return msg;



Dashboard

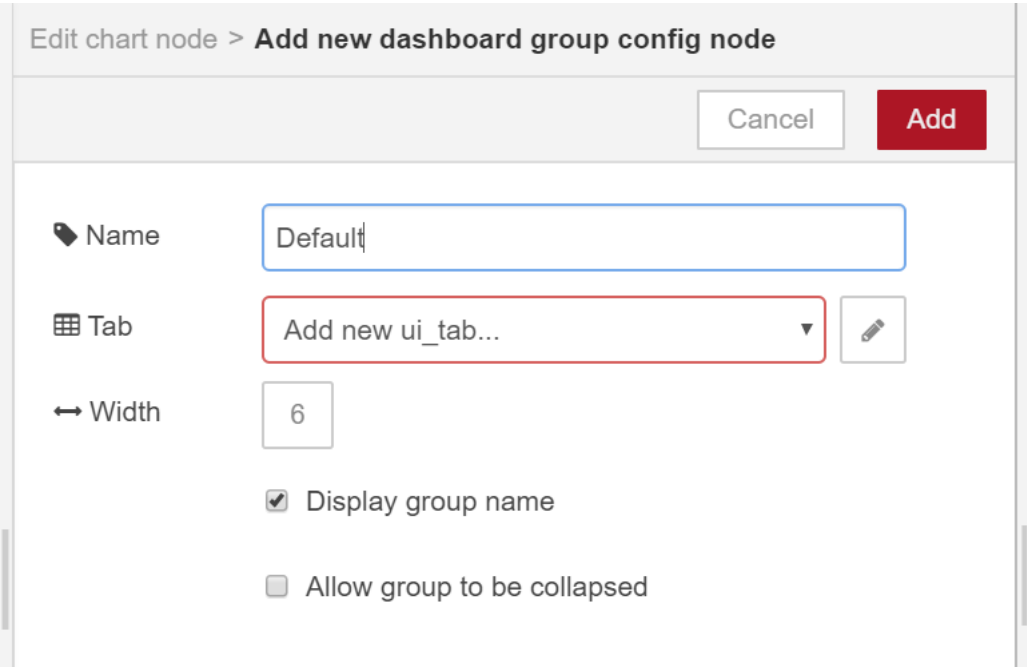
- Add a chart node 

and link it to our function node. Change the X-axis to last 5 minutes and click on the pencil next to the group field to configure the tabs of the UI.



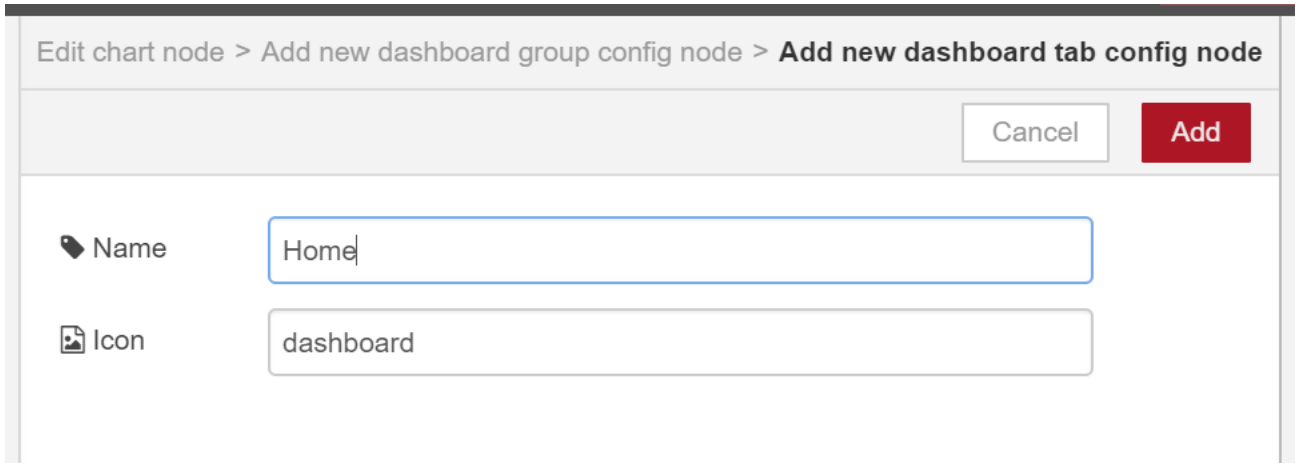
Dashboard

- The next screen appears. Click on the pencil next to the tab field.



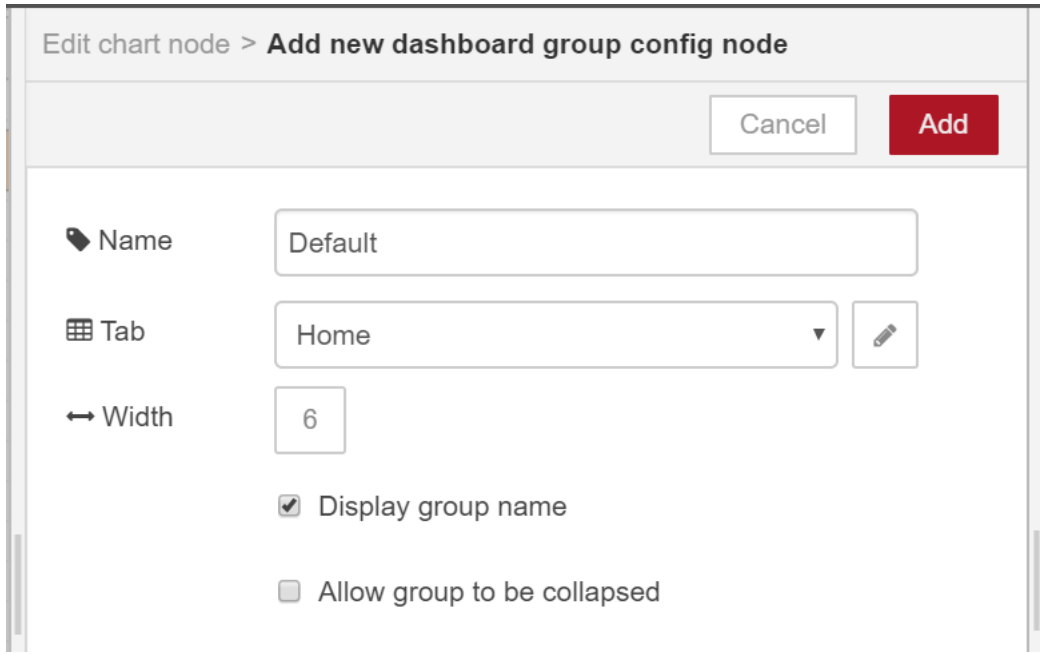
Dashboard

- Setup the tab config as home and click **Add.**



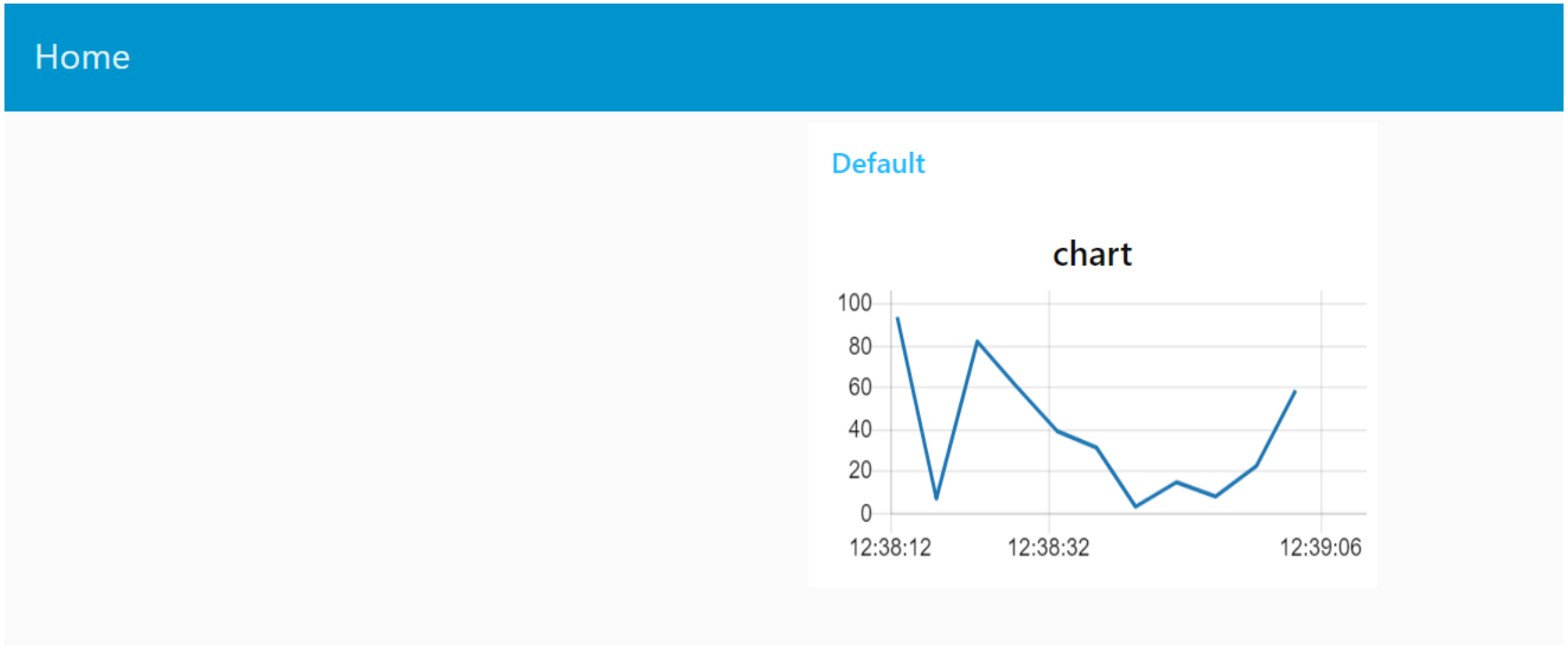
Dashboard

- On the add group config click **Add** and click **Done** to save the chart node.



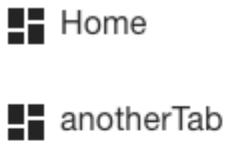
Dashboard

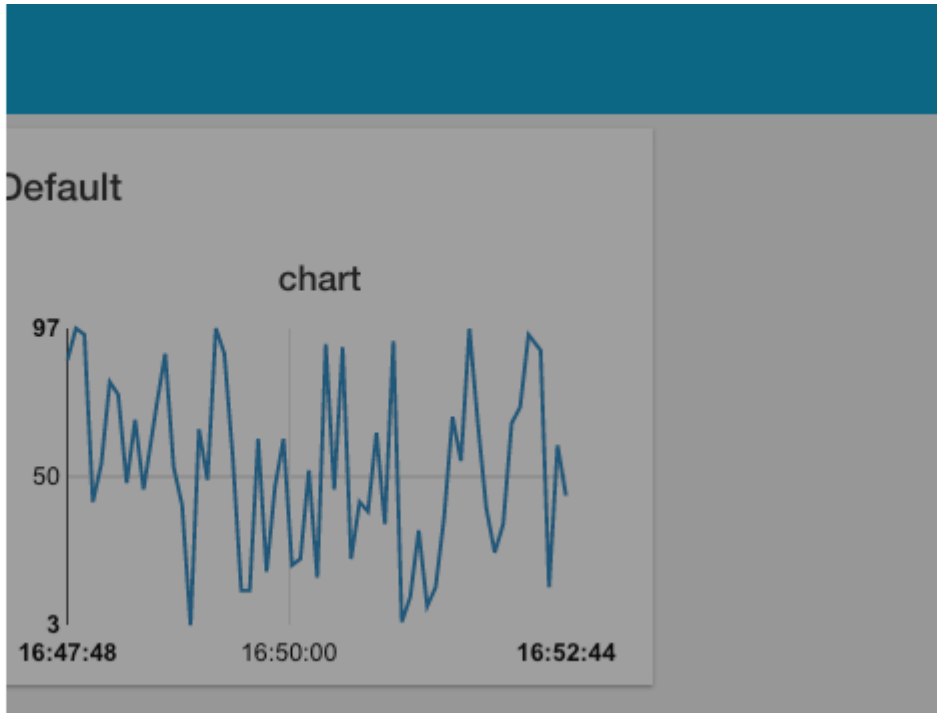
- Click **Deploy** and go to https://<yourinstancename>.eu-gb.mybluemix.net/ui/ and watch the chart change each 5 seconds.



Dashboard

- If you look at the top left of the web page, you can see that we are, by default, on the home tab. If you had created your own tab then when you click the selector top left you'll get a pull down menu of your tab options:



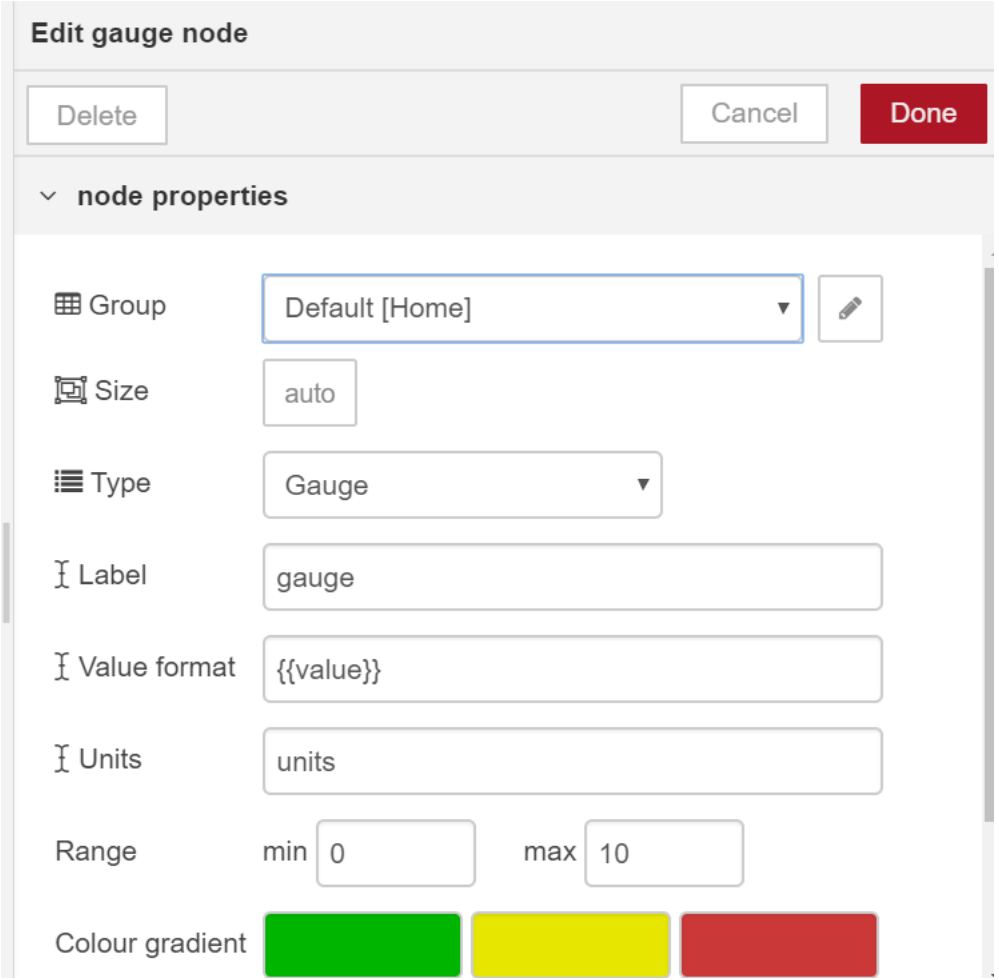


Dashboard

- Add a gauge node 

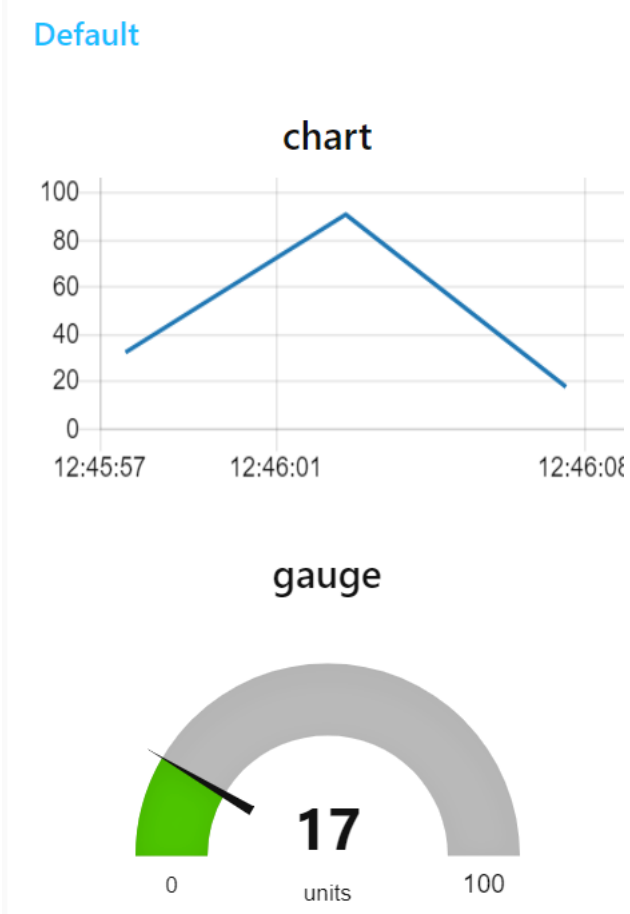
and also link it to our random number function node. We add it to our same group.

The Min and Max fields allow you to set the min and max values the gauge will shown. Make sure the max is set to 100 which is the most that the random number function node will generate. You can also change the ***Colour gradient*** to show different colours on the widget.



Dashboard

- Click **Deploy**. You should see the following screen.

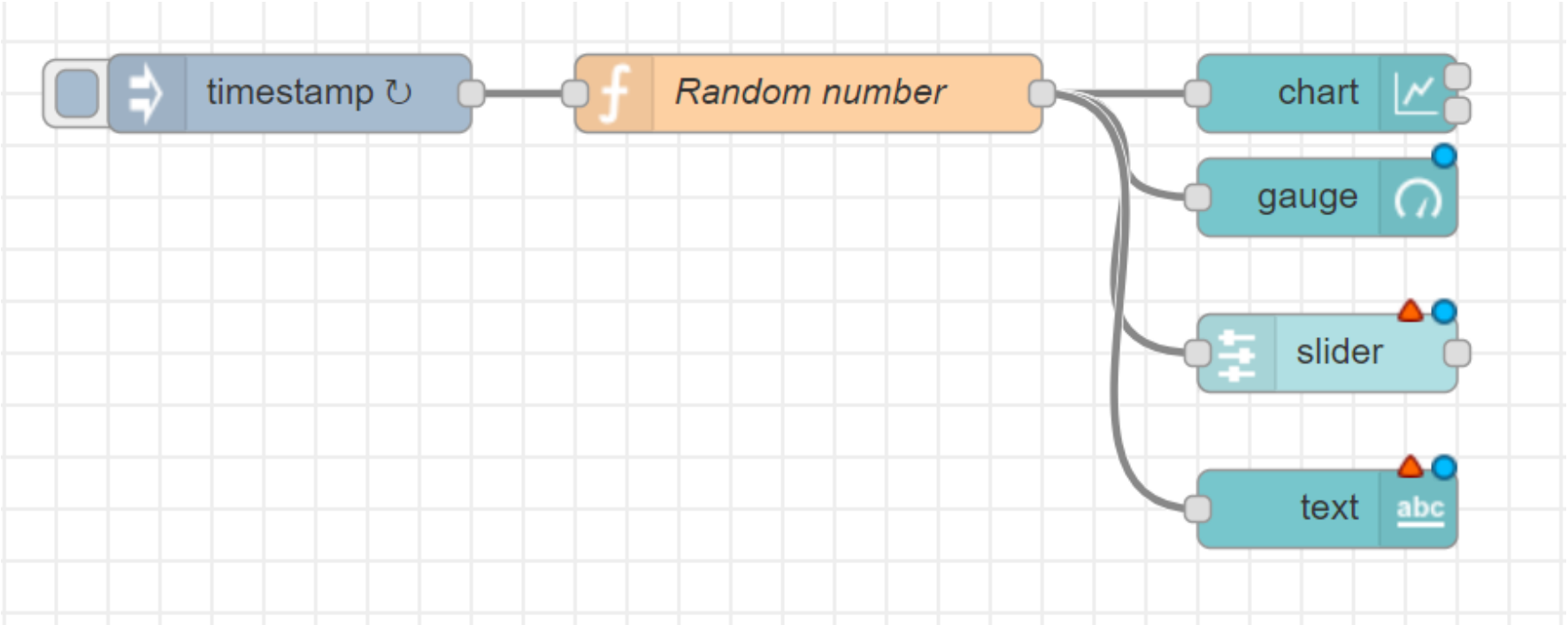


Dashboard

- Add a slider node 

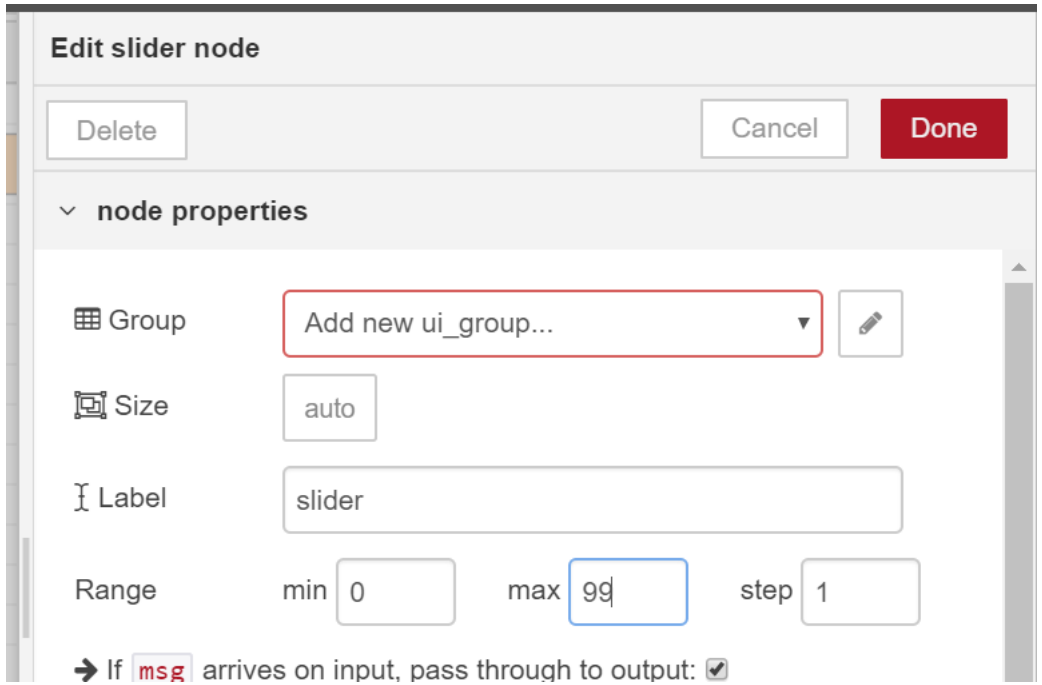
and link it to the random number function
- Add a text node 

and link it to the random number function



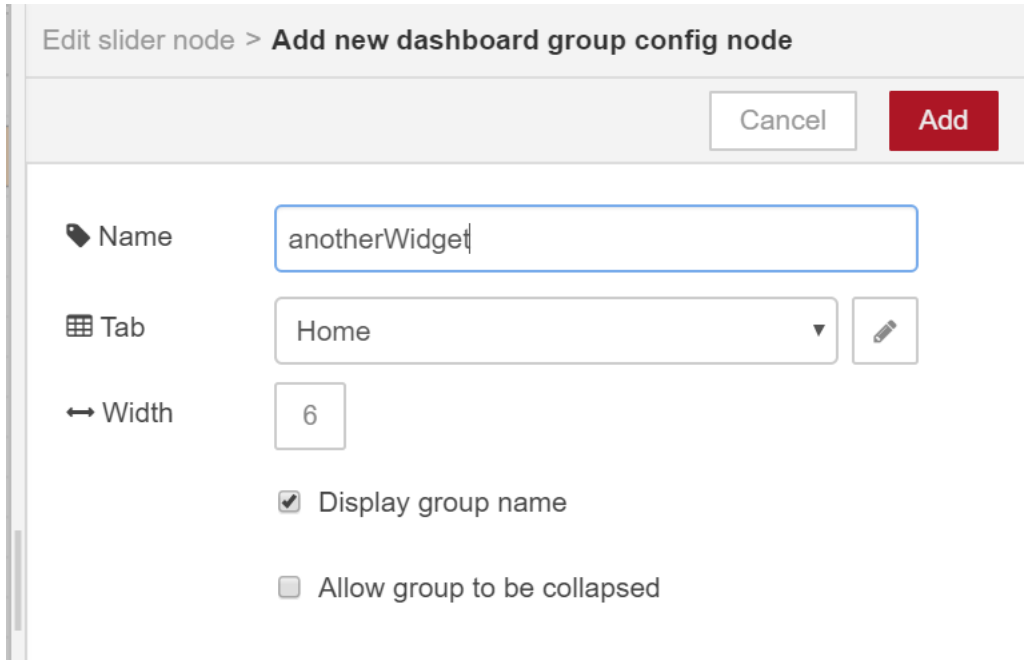
Dashboard

- For these two nodes, configure them to use the same tab – “Home” but use group name “anotherWidget”(You will need to click “Add new UI_group” from the drop down menu of the Group field, and then click the edit button).



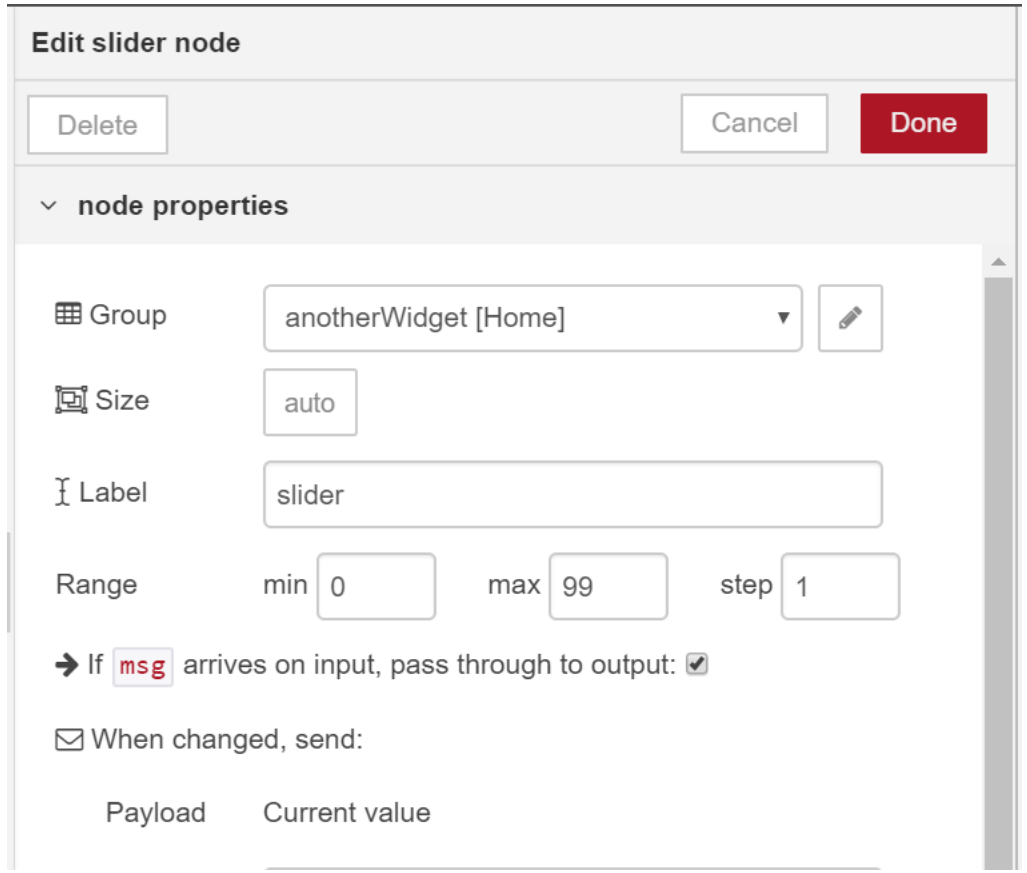
Dashboard

- Enter the anotherWidget name and click **Add.**



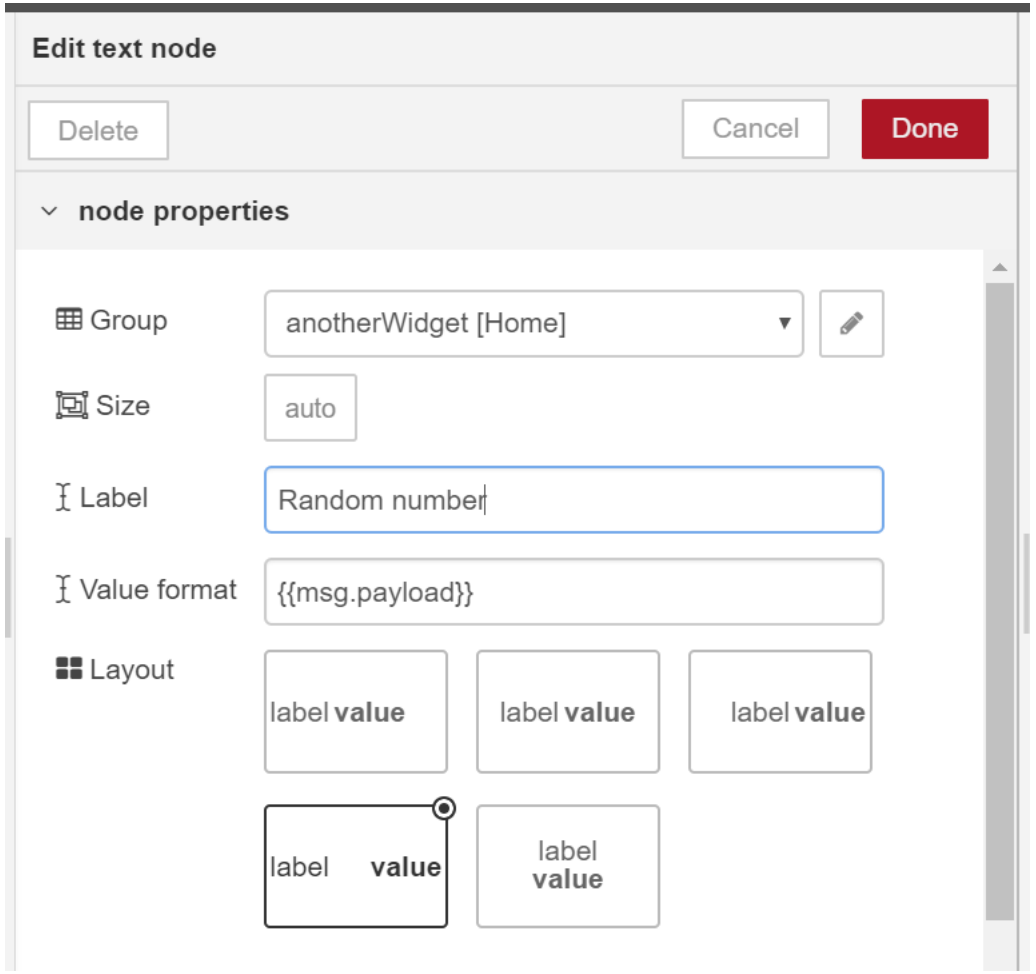
Dashboard

- Set the min/max values on the slider and click **Done**.



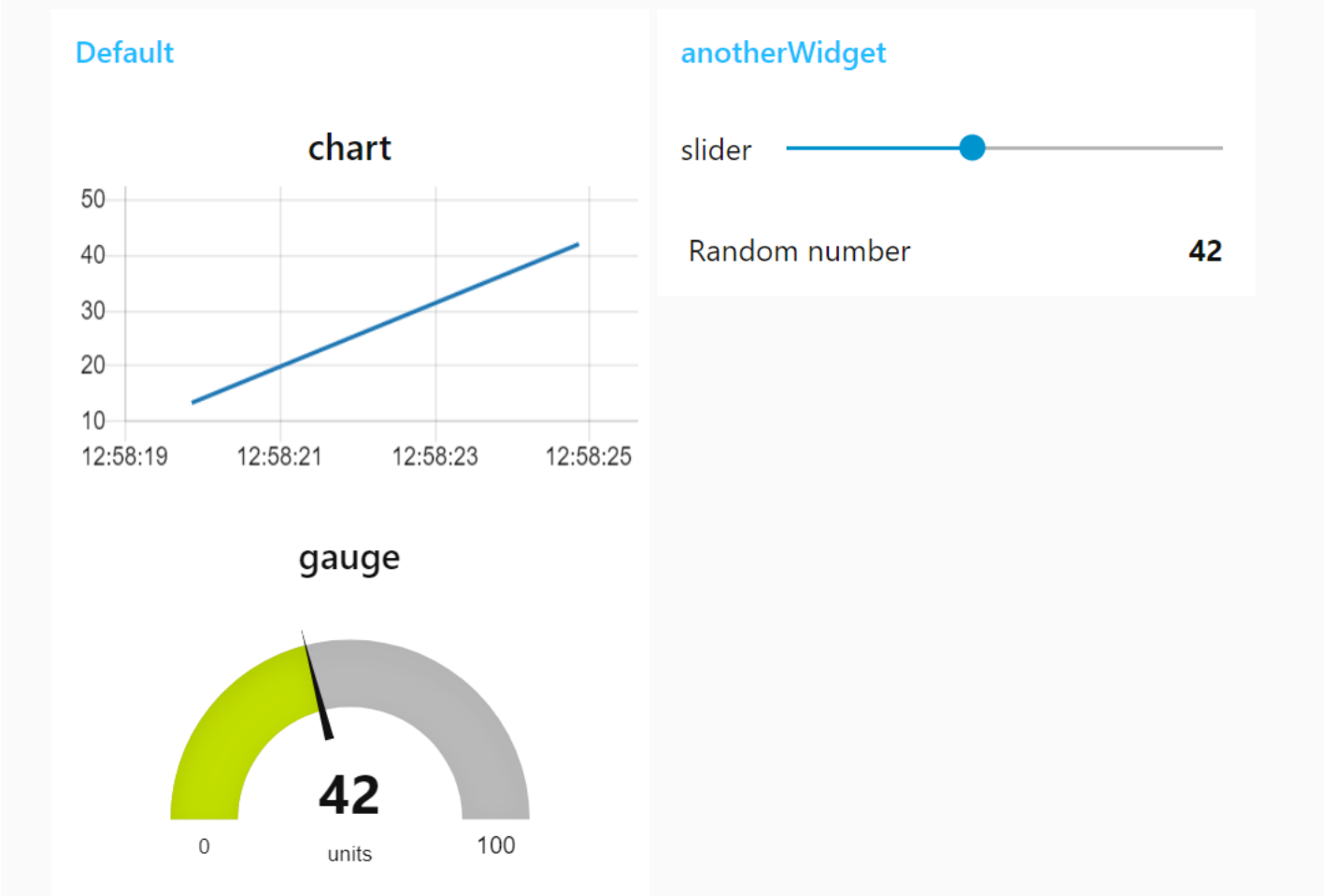
Dashboard

- Set up the text node as follows :



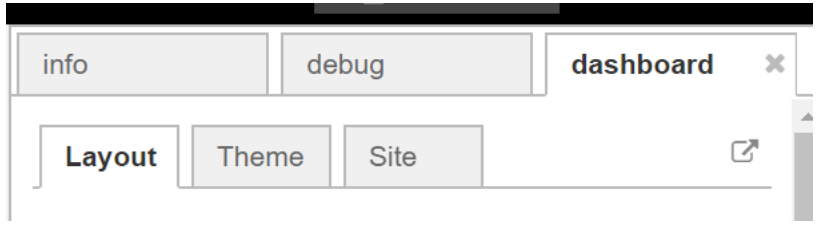
Dashboard

- Click **Deploy**. You should see the following screen.



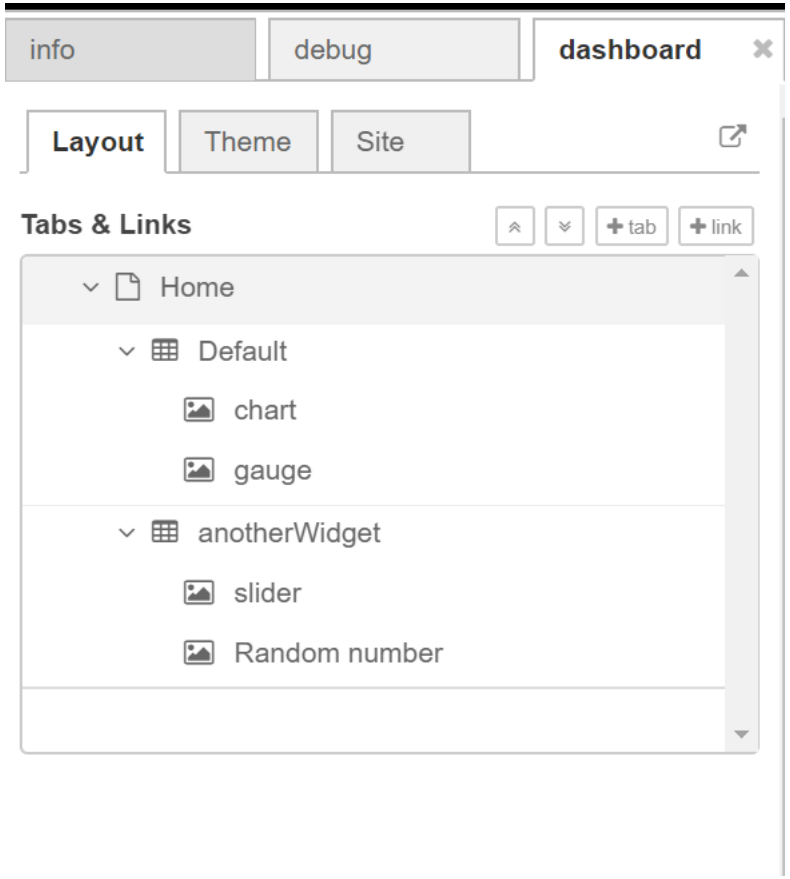
Dashboard

- In the dashboard tab beside your debug tab, you can also set the theme and order of the elements.



- If you don't see the dashboard tab, click the menu button at top right corner, then select "View" -> "Dashboard".
- You can see all the widgets and tabs showing in a tree structure,
- and you can easily drag the elements to change the orders that they are presented in the dashboard.

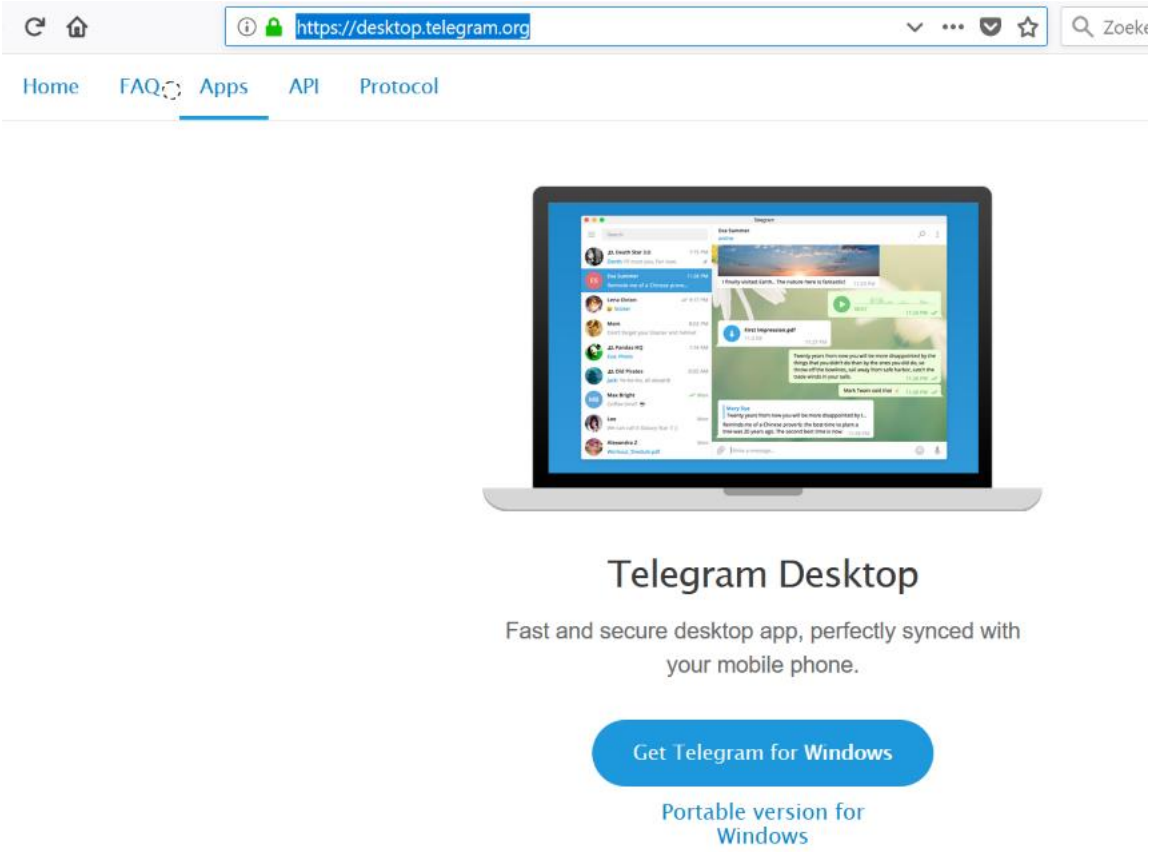
Dashboard


info
debug
dashboard
Layout
Theme
Site
Tabs & Links
tab
link
Home
Default
chart
gauge
anotherWidget
slider
Random number

CHATBOT

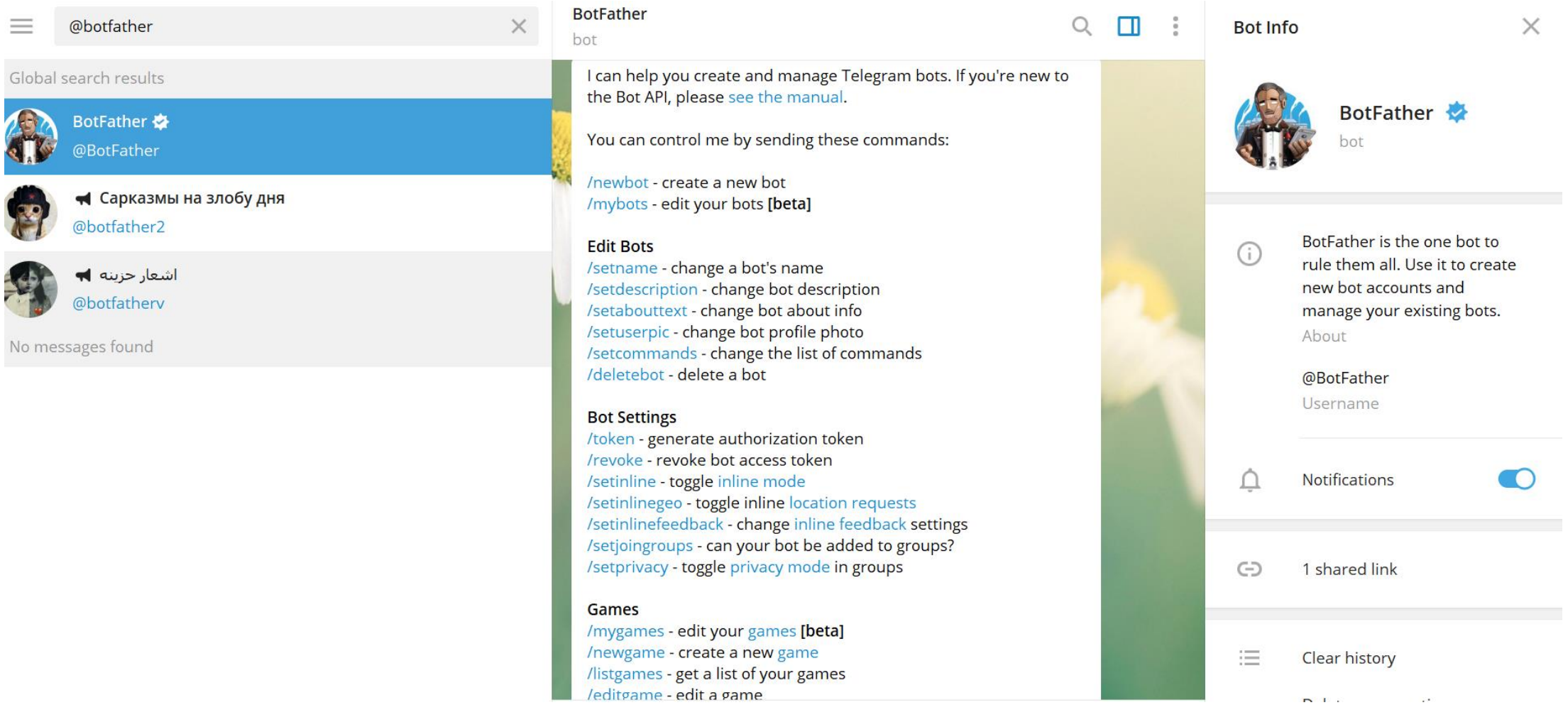
Chatbot

- **Create a new bot on Telegram's BotFather**
- Download and start the telegram app and create a free account. (Telegram.org)



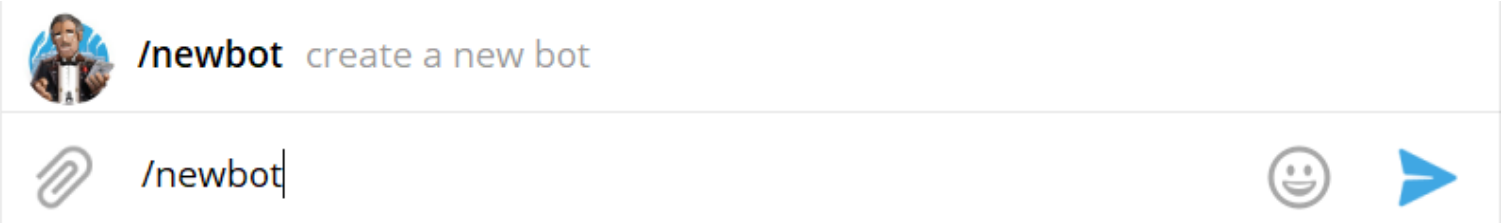
Chatbot

- Search for **@BotFather** at the search bar on top and select it



Chatbot

- send **/newbot** command / message to **BotFather**



- Enter the name and username of your bot, for example: name: **Watson Chat Bot**
- username: **WatsonChat<your_initials>Bot** (Replace <your_initials> with your initials or any other names) (info WatsonChatKDCBot)

Chatbot

- Once created, you'll be given a token string

Done! Congratulations on your new bot. You will find it at t.me/WatsonChatKDCBot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

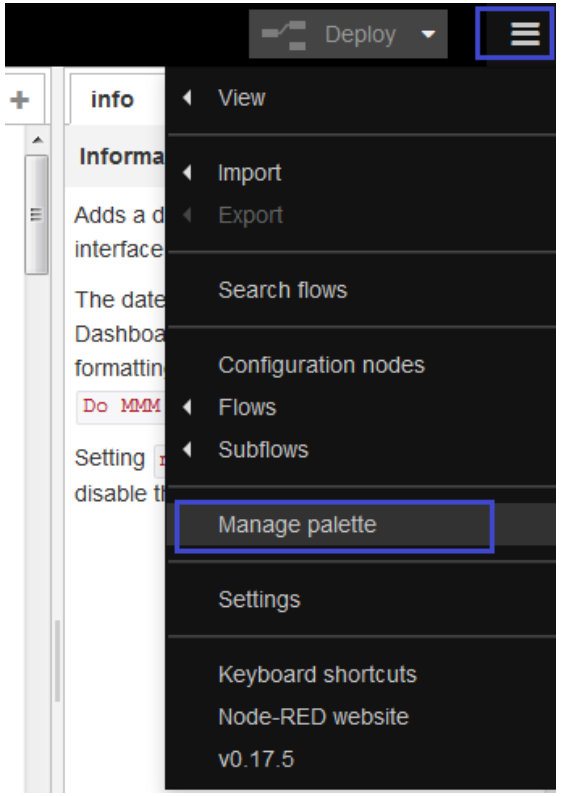
Use this token to access the HTTP API:
`374024329:AAEUX8oCPlgmPOxT78yDD8V03x-0I2Egnwk`

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api

14:50

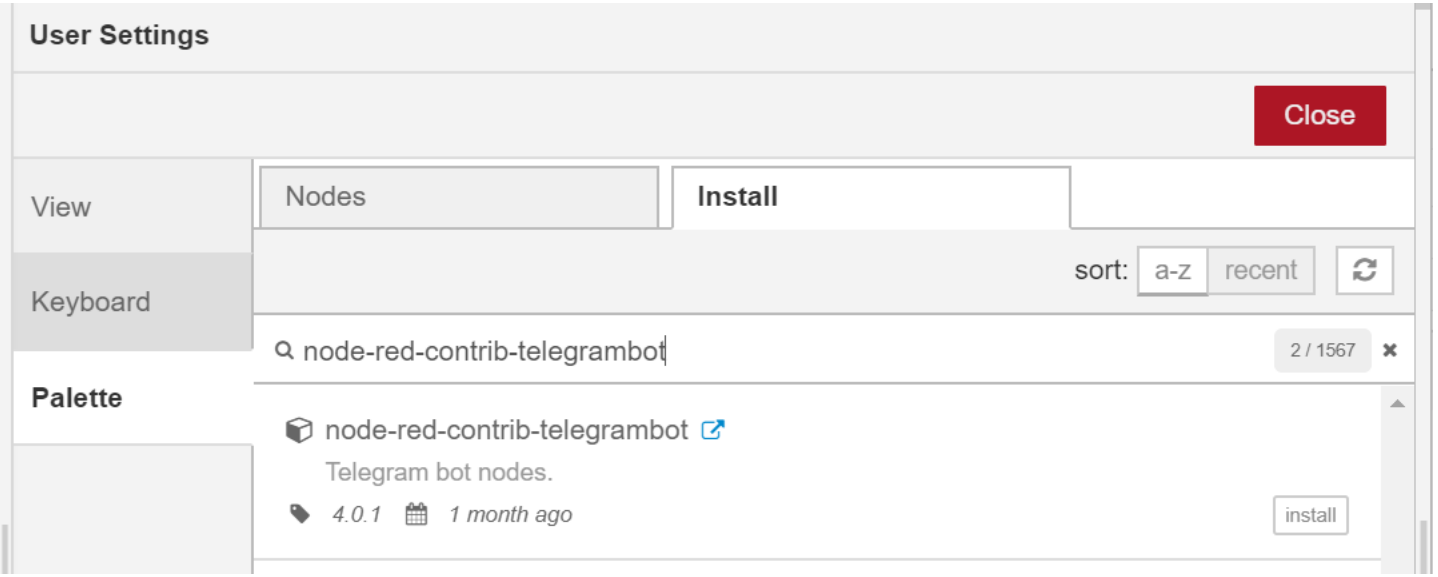
Chatbot

First, we need to install the *node-red-contrib-telegrambot* node in our Node-RED palette. In Node-RED, click the button at the upper-right corner and click **Manage palette**.



Chatbot

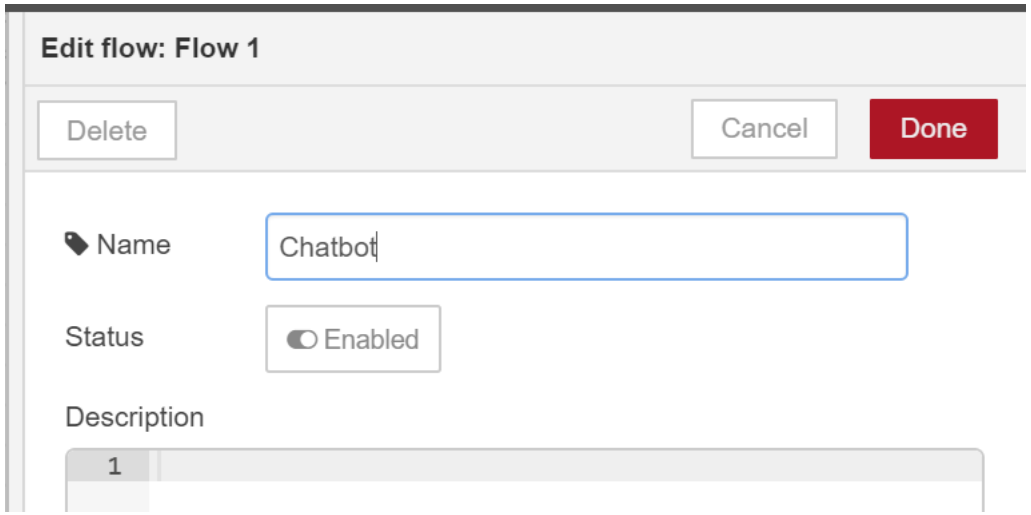
- On the **Install** tab, search for *node-red-contrib-telegrambot*, and click **Install**. Wait for the installation confirmation dialog box to be displayed, and then restart Node-RED if necessary.



Chatbot

- Create a new flow tab by clicking +.

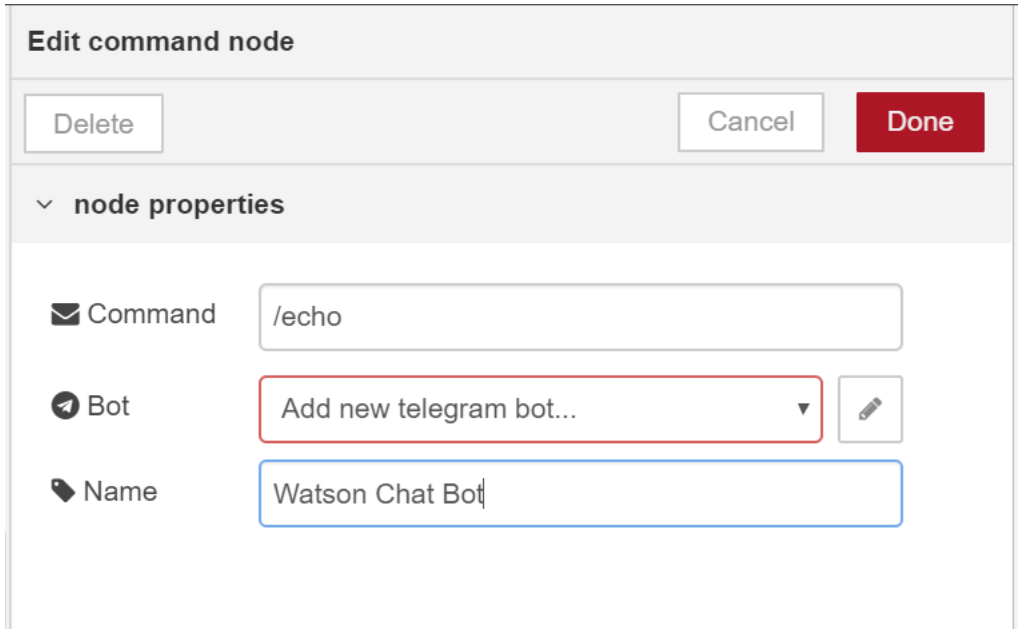
- Double-click the new tab and enter a name for the new flow tab. Then click **Done. If the edit screen does not appear click the right menu and rename.**



Chatbot

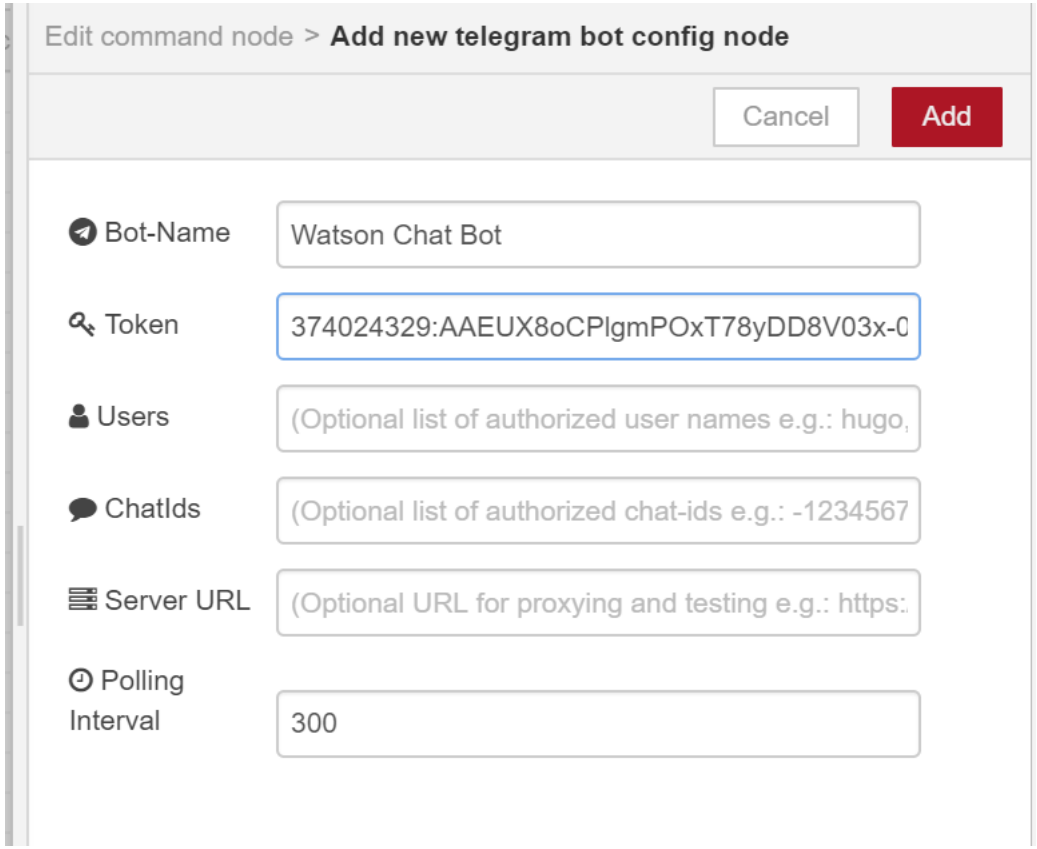
- 

ıg a telegram command node on the canvas. Enter /echo in the command, change the name to Watson Chat Bot. Click on the pencil to edit the chatbot settings.



Chatbot

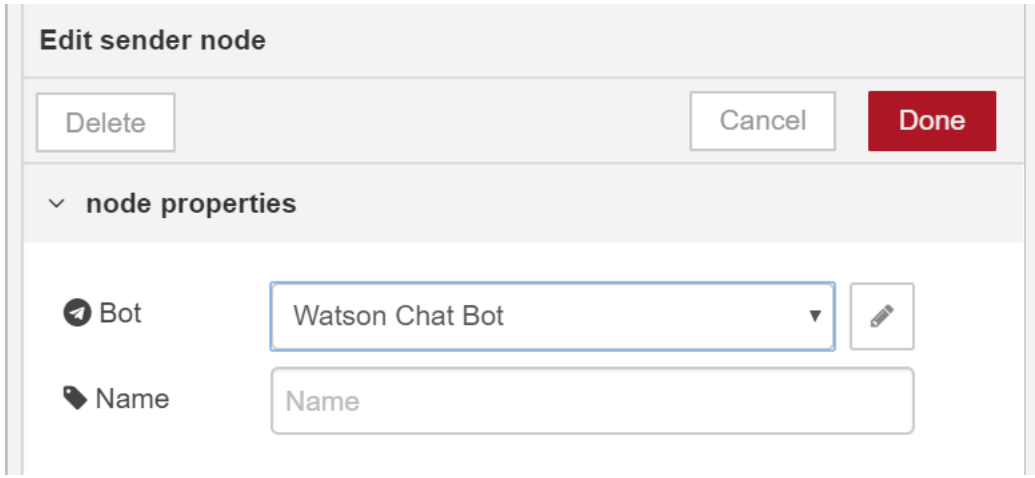
- Fill in name and Token. Leave other fields blank and click Add and done.



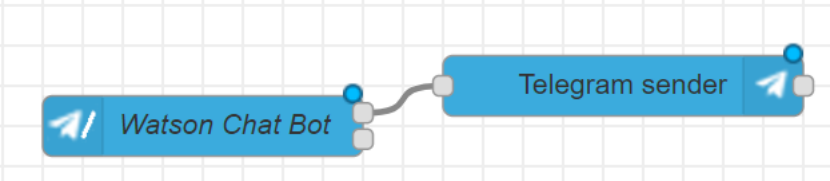
Chatbot

- 

d a telegram sender node. Link it to the top output of the command node. Double click it. Select the bot you just defined and click done.

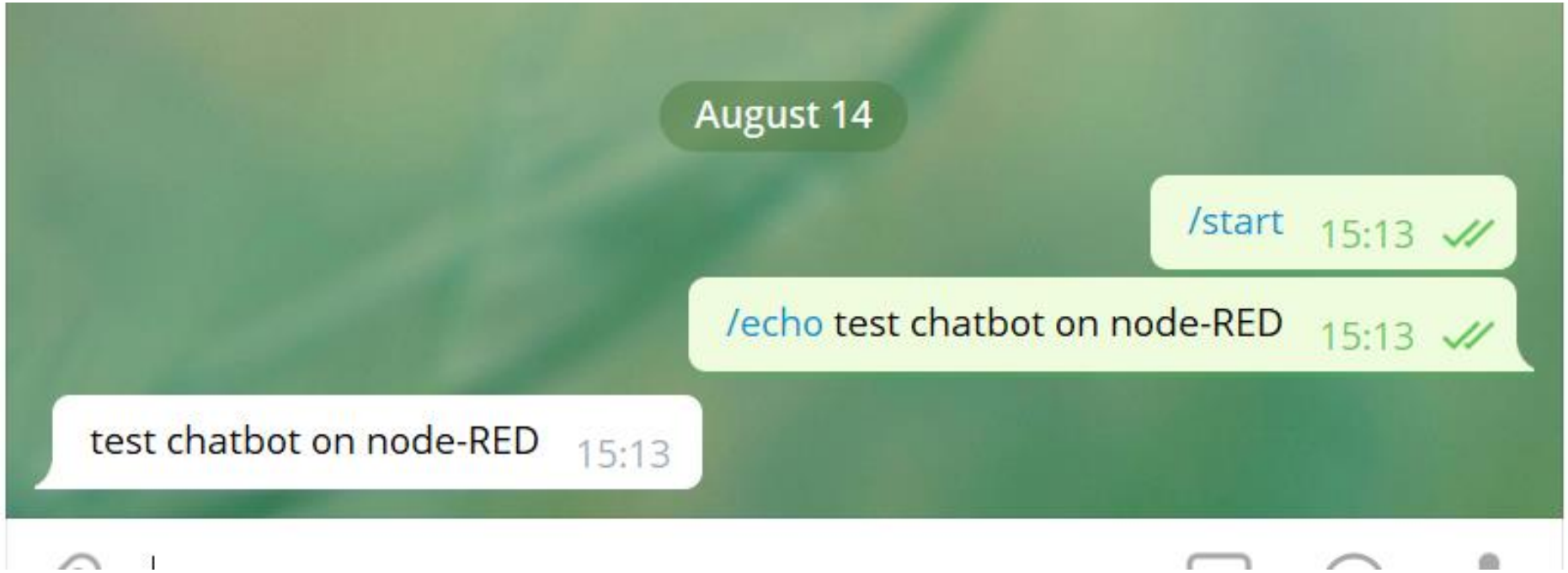


- Your flow should look like this. Click Deploy.



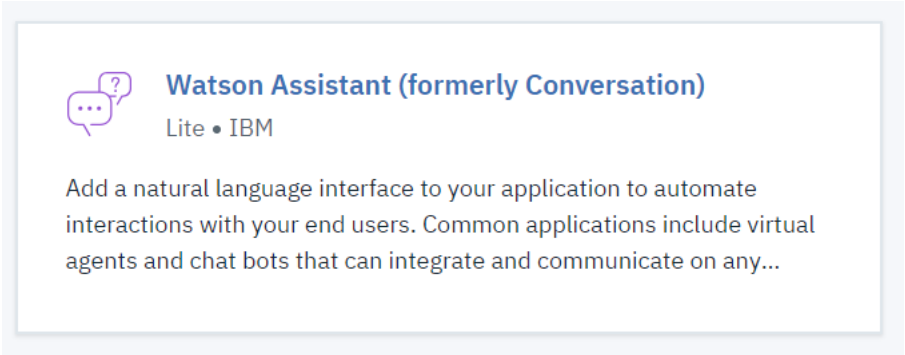
Chatbot

- On your **Telegram** app add the bot that you have created as a contact
- Send the following message: /echo <any_message_here>
- You should receive the same message back, without the **/echo** command



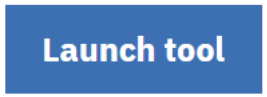
Chatbot

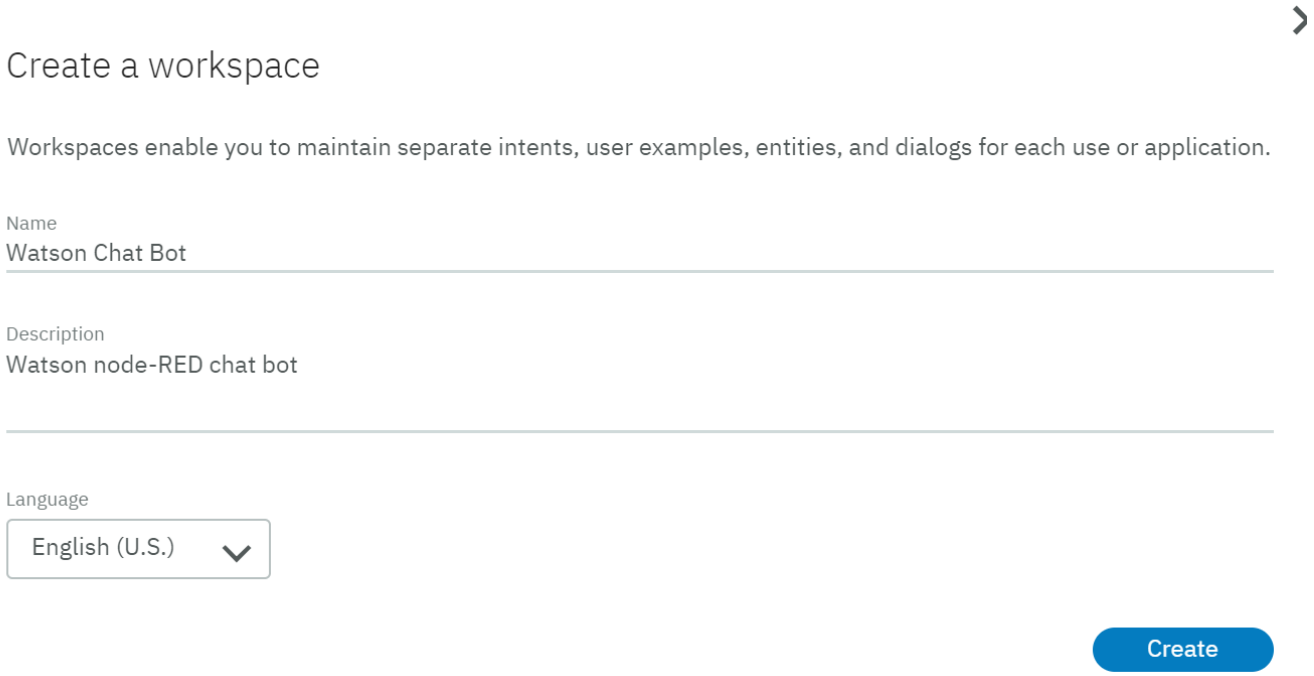
- Go to the IBM Cloud catalog, and search for **Watson Assistant**.
- Select the service and ensure that the region, organization, and space are the same as your Node-RED instance.
- Click **Create** to create the service.



Chatbot

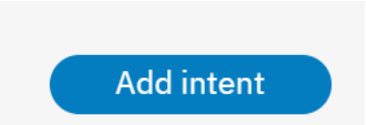
- Click launch tool to start the tool.
- Create a new workspace by clicking **Create**
- Enter **Watson Chat Bot** as the **Name** and click **Create**



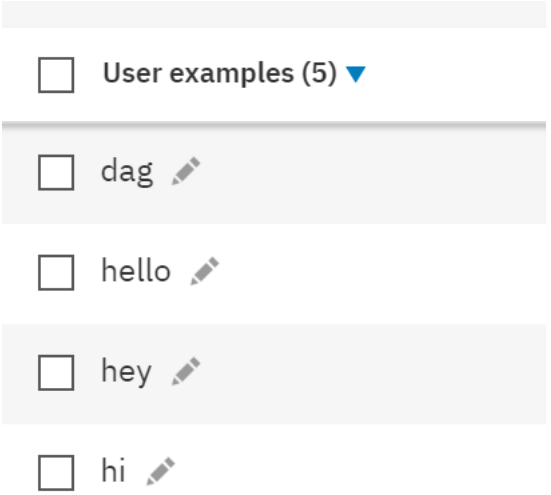


Chatbot

- Click on add intent



- Enter Intent name: #greetings and click create intent now add the following user examples : hello, hi, howdy, hey, dag



Chatbot

- Create another one with the following values:

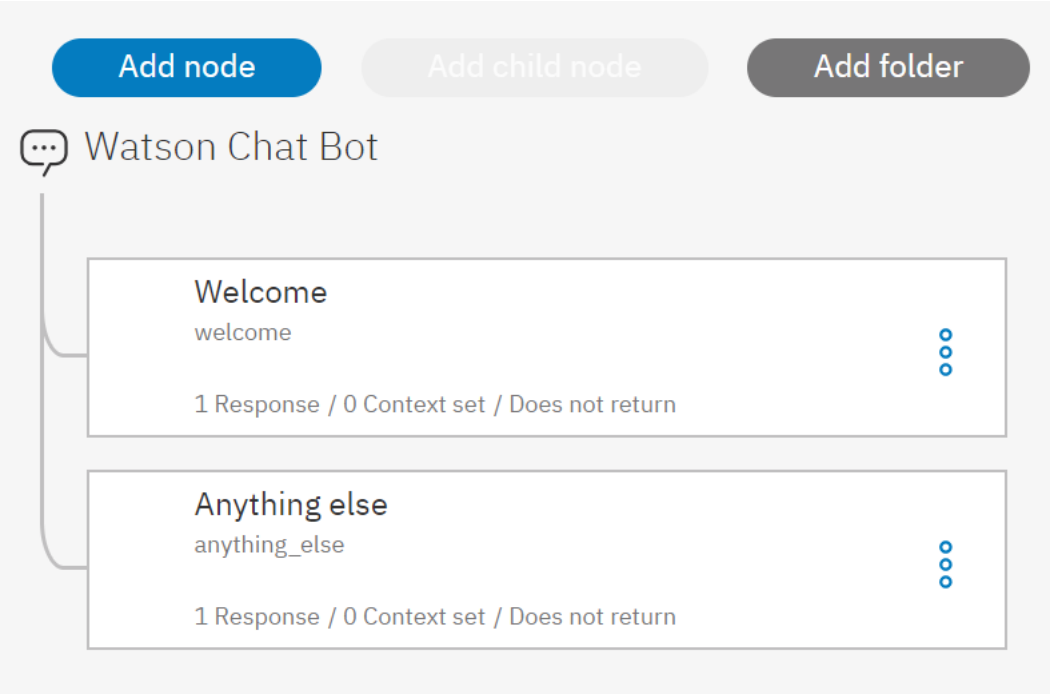
Intent name: #farewell

User example:

- goodbye
- bye
- so long
- see you later
- ciao

Chatbot

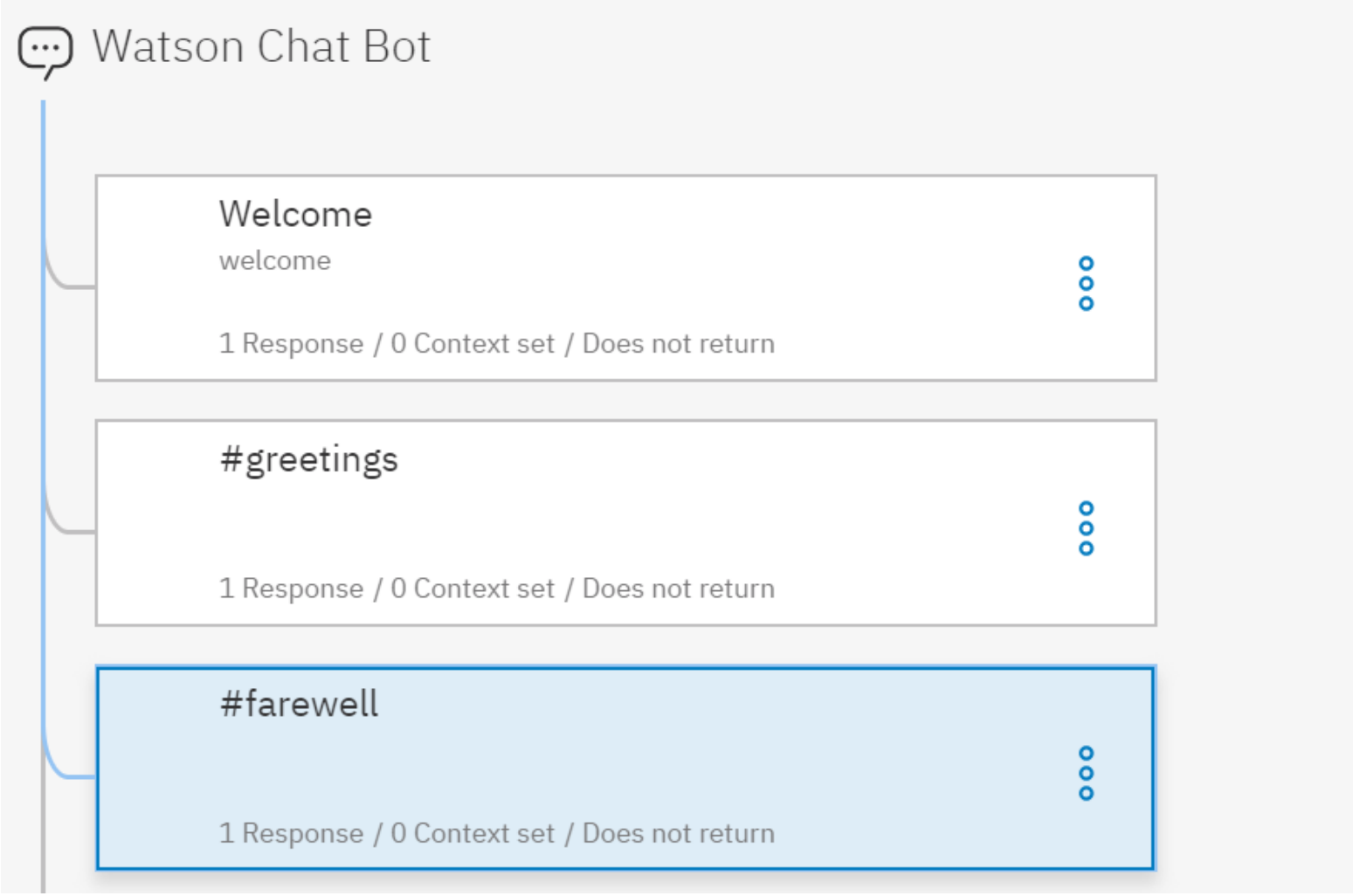
- Click on the Dialog tab to configure the dialog flow
- Click Create
- Click add node button



Chatbot

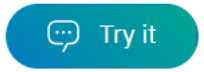
- Name this node: Greetings
- If bot recognizes: #greetings
- Then respond with: hello
- Then, click the button on the right of the Greetings node and click Add node below to create another node.
- Name this node: Farewells
- If bot recognizes: #farewell
- Then respond with: goodbye

Chatbot

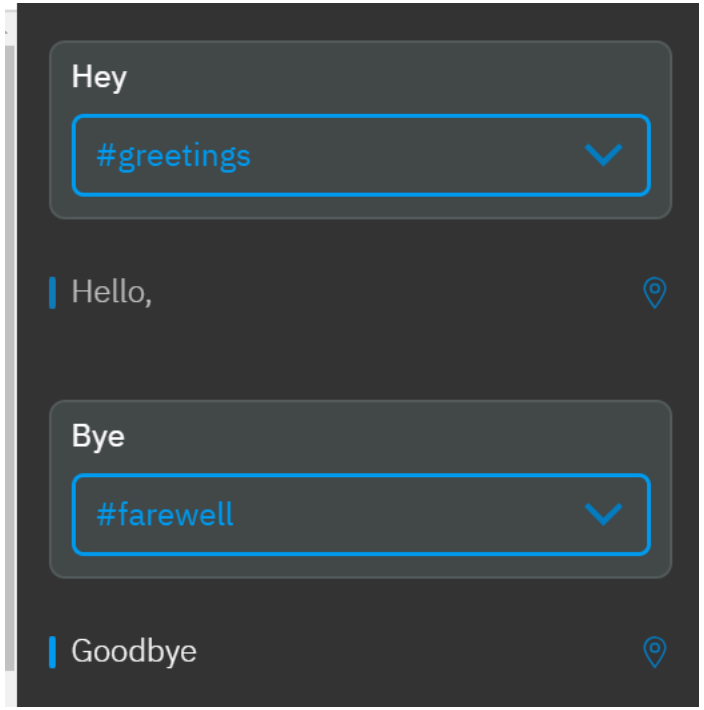

Watson Chat Bot
Welcome
welcome
1 Response / 0 Context set / Does not return
#greetings
1 Response / 0 Context set / Does not return
#farewell
1 Response / 0 Context set / Does not return

Chatbot

- To test it, click on the try it icon at the top-right, to open a chat box

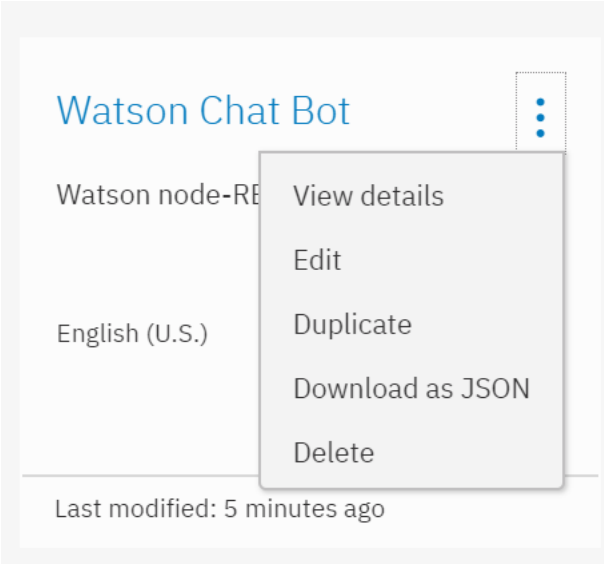


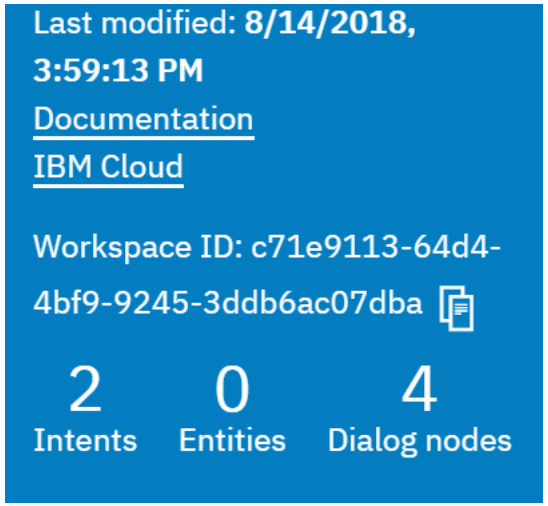
- Try saying the user examples that we created earlier, and it should response accordingly
- Even with some minor typos, or missing words, it may recognize your intent and response accordingly



Chatbot

- Click workspaces, click view details. Copy the workspace ID for use in node-RED





Chatbot

- Add another flow with the following connection sequence of nodes and connect the output of the node to the input of the next node: **telegram command- function - assistant - function - telegram sender**

- Configure both telegram nodes to use the **WatsonChatBot** and set the Telegram Command to use the **/watson** command

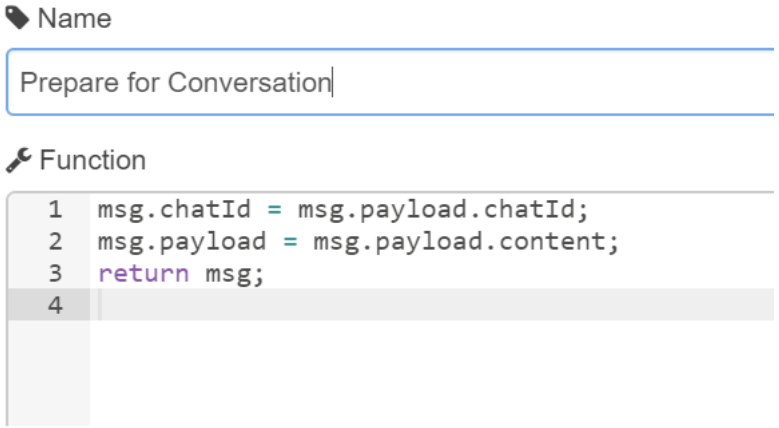
Chatbot

- Double click the 1st function node to configure it
- Enter *Prepare for Conversation* as the Name
- Enter the following as the Function and click Done

msg.chatId = msg.payload.chatId;

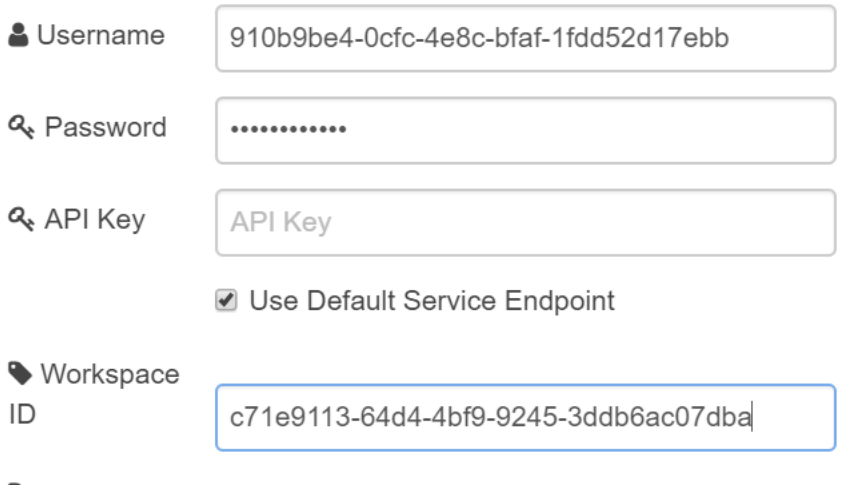
msg.payload = msg.payload.content;

return msg;



Chatbot

- Double click the Assistant node to configure it
- Enter your credentials
- Enter the Workspace ID that you have copied earlier, then click Done



Chatbot

- Double click the 2nd Function node to configure it
- Enter *Prepare for Telegram* as the Name
- Enter the following as the Function and click Done

```
msg.payload = {
  chatId : msg.chatId,
  type : "message",
  content : msg.payload.output.text[0]};
return msg;
```

Name

Prepare for Telegram

Function

```
msg.payload = {
  chatId : msg.chatId,
  type : "message",
  content : msg.payload.output.text[0]};
return msg;
```

Chatbot

- Click Deploy
- On your Telegram app send the following message to the chat bot: /watson hey
- You should receive **hello** message back
- Try with farewell messages instead
- You should receive **goodbye** message back

Chatbot

- Go to the Watson assistant and create a new intent with the following values:

Intent name: # translate

User example:

- Can you translate?
- Please translate
- Translate
- Do you know how to translate?
- Translate for me please

Chatbot

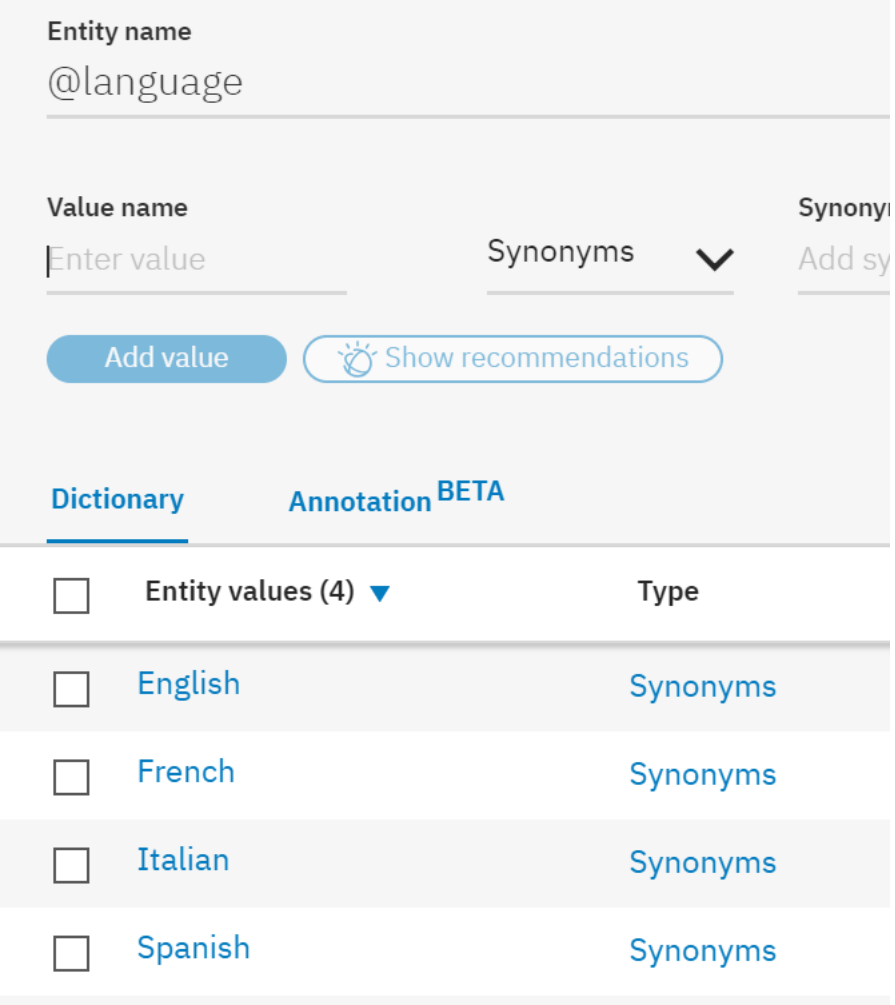
Click on the **Entities** tab and click the **Create new** button

- Enter the following and click **Create**:
- Entity name: **language**

Values:

- English
- French
- Italian
- Spanish

Chatbot


Entity name
@language
Value name
Enter value
Synonyms
Add value
Show recommendations
Dictionary
Annotation BETA
Entity values (4)
Type
English
Synonyms
French
Synonyms
Italian
Synonyms
Spanish
Synonyms

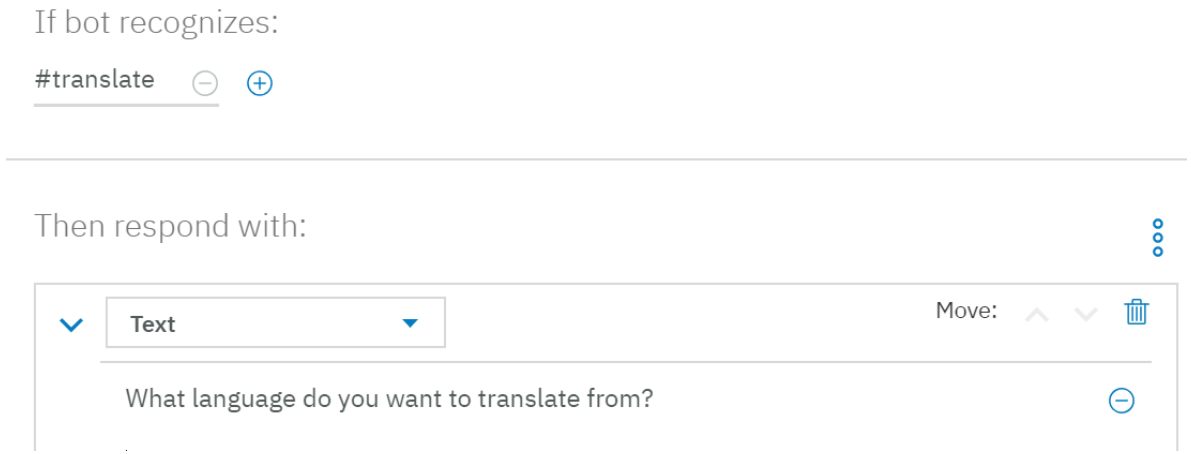
Chatbot

- Click on the Dialog tab and add a new dialog node under #farewell
- Enter the following and click the X button:

Name this node: Translate

If bot recognizes: #translate

Then respond with: What language do you want to translate from?



Chatbot

- Add a new child node under #translate
- Enter Source Language as the Name, @language on the “If bot recognizes”, , click on the button on the right under Then respond with, and click Open JSON Editor

Enter the following text:

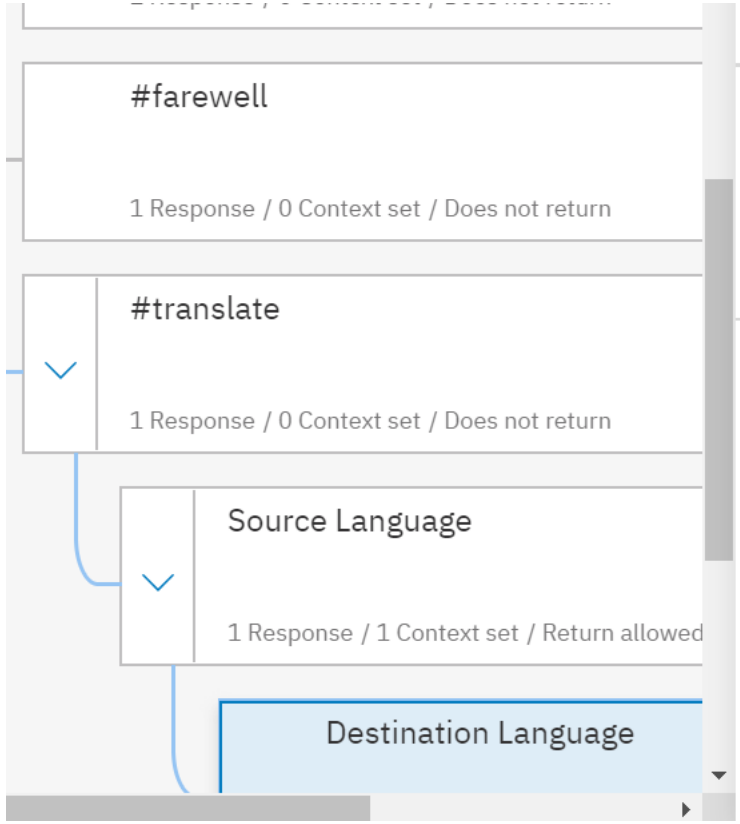
```
{
 "context": { "source": "@language" },
 "output": {
  "text": {
   "values": [ "What language do you want to translate to?" ]}}}
```

Chatbot

- Add another child node under source language
- Enter Destination Language as the Name, @language on the “If bot recognizes”, , click on the button on the right under Then respond with, and click Open JSON Editor
- Enter the following text:

```
{
 "context": { "destination": "@language" },
 "output": {
  "text": {
   "values": [ "What is the text that you want to translate?" ]}}}
```

Chatbot



If bot recognizes:

Destination Language

Then respond with:

```
{
  "context": { "destination": "@language" },
  "output": {
    "text": {
      "values": [ "What is the text that you want to translate?" ]}}}
```

Chatbot

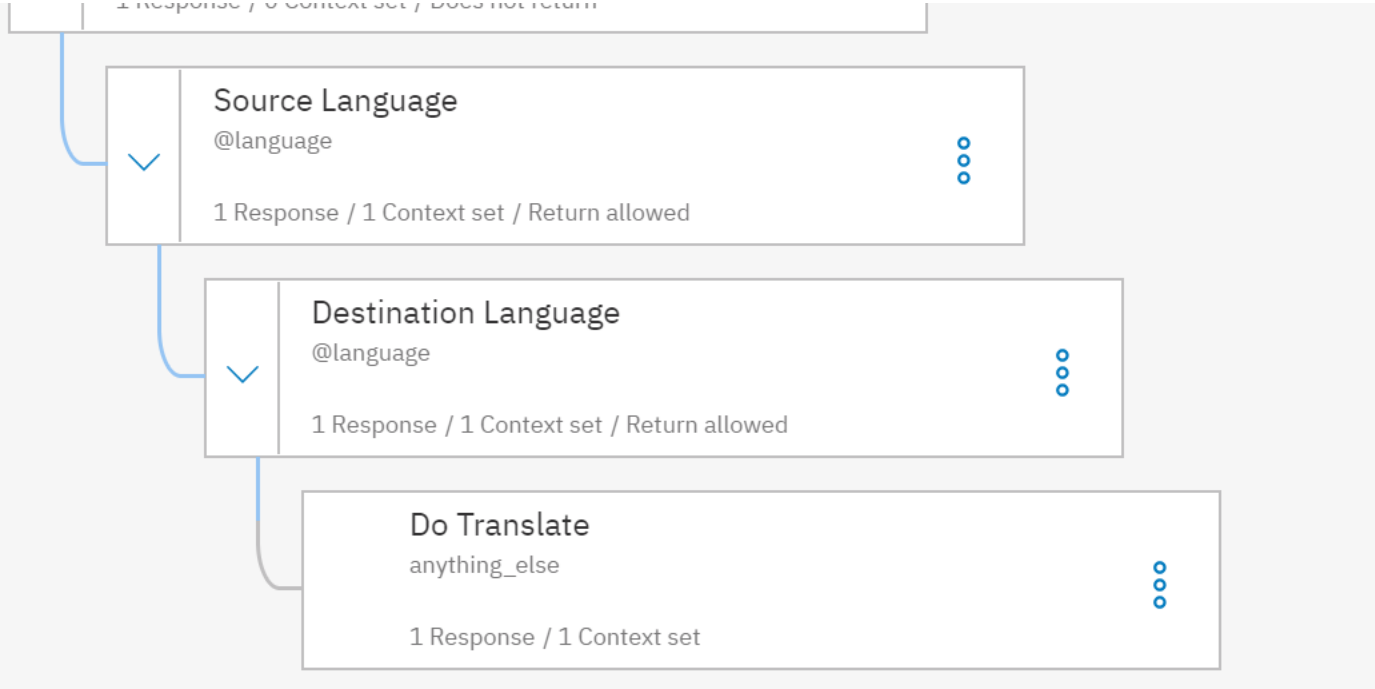
- Add another child node under destination language

Enter Do Translate as the Name, anything_else on the “If bot recognizes”, click on the button on the right under Then respond with, and click Open JSON Editor

Enter the following text:

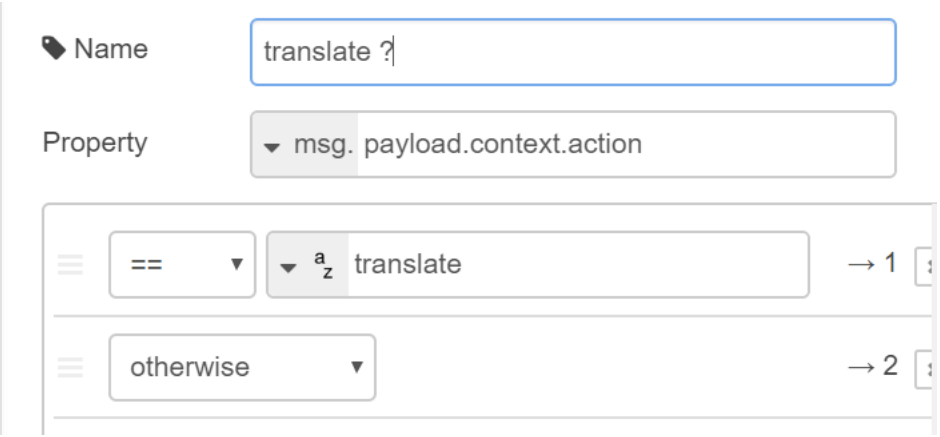
```
{
  "context": { "action" : "translate" },
  "output": {}
}
```

Chatbot


Source Language
@language
1 Response / 1 Context set / Return allowed
Destination Language
@language
1 Response / 1 Context set / Return allowed
Do Translate
anything_else
1 Response / 1 Context set

Chatbot

- Go back to the Node-RED workspace
- Add a switch node and position it at the right of the assitant node
- Double click the switch node and set the Property to msg.payload.context.action
- At the option below it, select == and enter translate as the value
- Click +add and select otherwise then click Done



Chatbot

- Add a function node next to the switch node
- Double click the function node to configure it
- Enter Prepare for Translator as the Name
- Enter the following as the Function and click Done

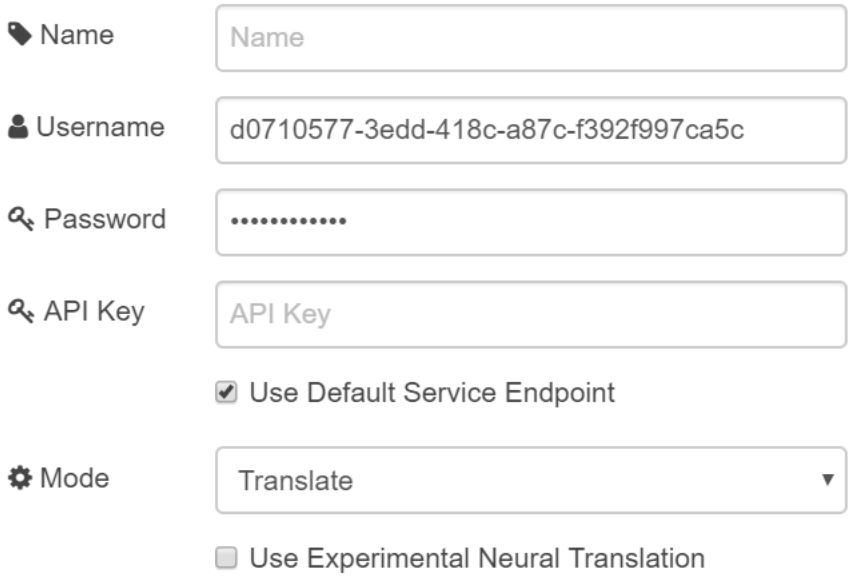
```
msg.srclang = getLanguage(msg.payload.context.source);
msg.destlang = getLanguage(msg.payload.context.destination);
msg.payload = msg.payload.input.text;

return msg;
```

Chatbot

```
function getLanguage(lang) {
  switch (lang.toLowerCase()) {
    case "french":
      return "fr";
    case "italian":
      return "it";
    case "spanish":
      return "es";
  }
  return "en";
}
```

Chatbot

- Next, add the language translator node
- Double click the language translator node to configure it



Chatbot

- Next, add another function node, name it as Prepare for Telegram 2 and put it to the right of the language translator node
- Enter the following as the Function and click Done

```
msg.payload = {
 chatId : msg.chatId,
 type : "message",
 content : msg.payload
};
return msg;
```

Chatbot

- Add one more function node next to the language translator node and name it Clear Conversation Context
- Enter the following as the Function and click Done

```
msg.payload = "hi";
msg.params = {
   context : {}
}
return msg;
```

Chatbot

- Copy the assistant node, put it next to the Clear Conversation Context function node and name it assistant 2
- Disconnect the connection between the conversation and Prepare for Telegram node
- Connect the following nodes accordingly
- assistant > switch
- switch (1st output) > Prepare for Translator
- Prepare for Translator > language translator
- language translator > Prepare for Telegram 2
- Prepare for Telegram 2 > Telegram Sender
- language translator > Clear Context
- Clear Context > assistant 2
- switch (2nd output) > Prepare for Telegram
- Then, click the Deploy button

Chatbot

- Test Language Translator Integration with Node-RED on Telegram
- On your Telegram app send the following message to the chat bot:

/watson please translate

You should receive a reply asking for the source language. Reply:

/watson english

Chatbot

- You should receive a reply asking for the target language. Reply:

/watson french

- You should receive a reply asking for the text to be translated. Reply with the text you want to translate, for example:

/watson Where is the nearest restroom?

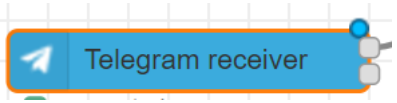
- You should receive a reply with the translated text

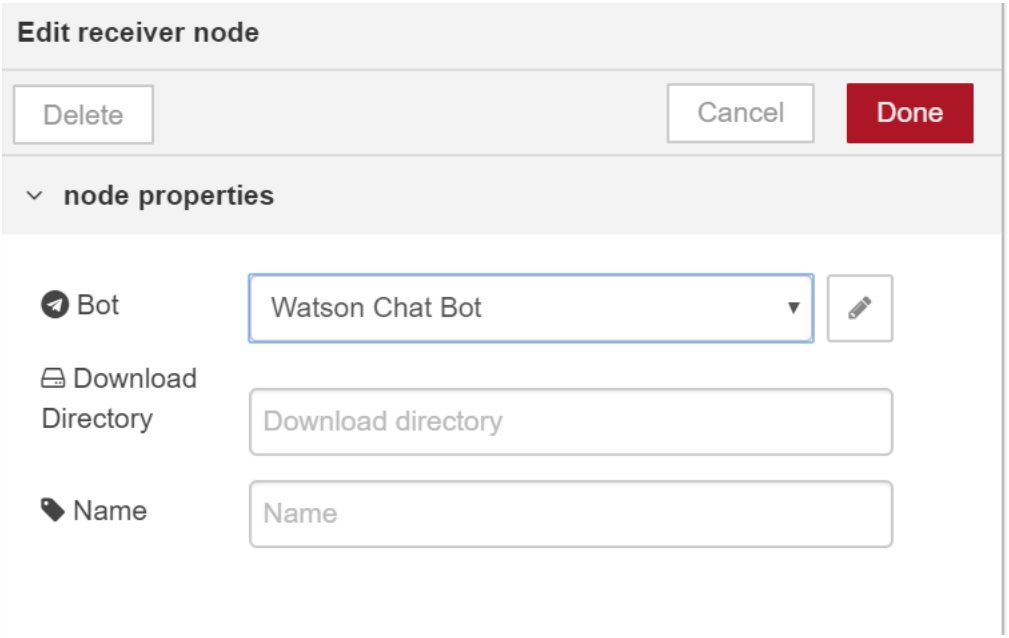
Chatbot


/watson translate 11:44
What language do you want to translate from? 11:44
/watson english 11:44
What language do you want to translate to? 11:44
/watson french 11:44
What is the text that you want to translate? 11:44
/watson Hello, how are you feeling today 11:44
Bonjour, comment vous sentez-vous aujourd'hui 11:44

Chatbot

- Change the telegram command node with a reciever node. You can now do the same thing without adding the /watson command to it.





Chatbot

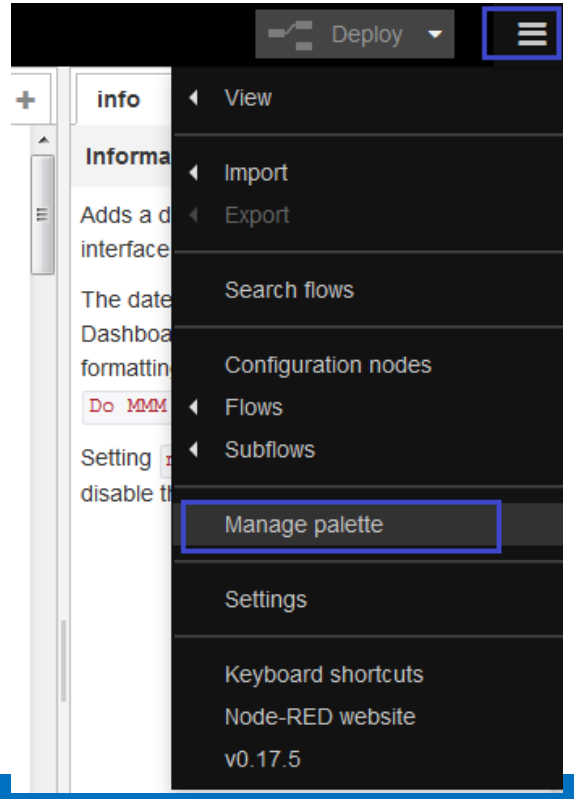

hey 11:48
Hello, 11:48
translate 11:48
What language do you want to translate from? 11:48
english 11:48
What language do you want to translate to? 11:48
french 11:48
What is the text that you want to translate? 11:48
Where is the bathroom ? 11:49
Où est la salle de bain? 11:49

DB2 INTEGRATION

DB2 integration

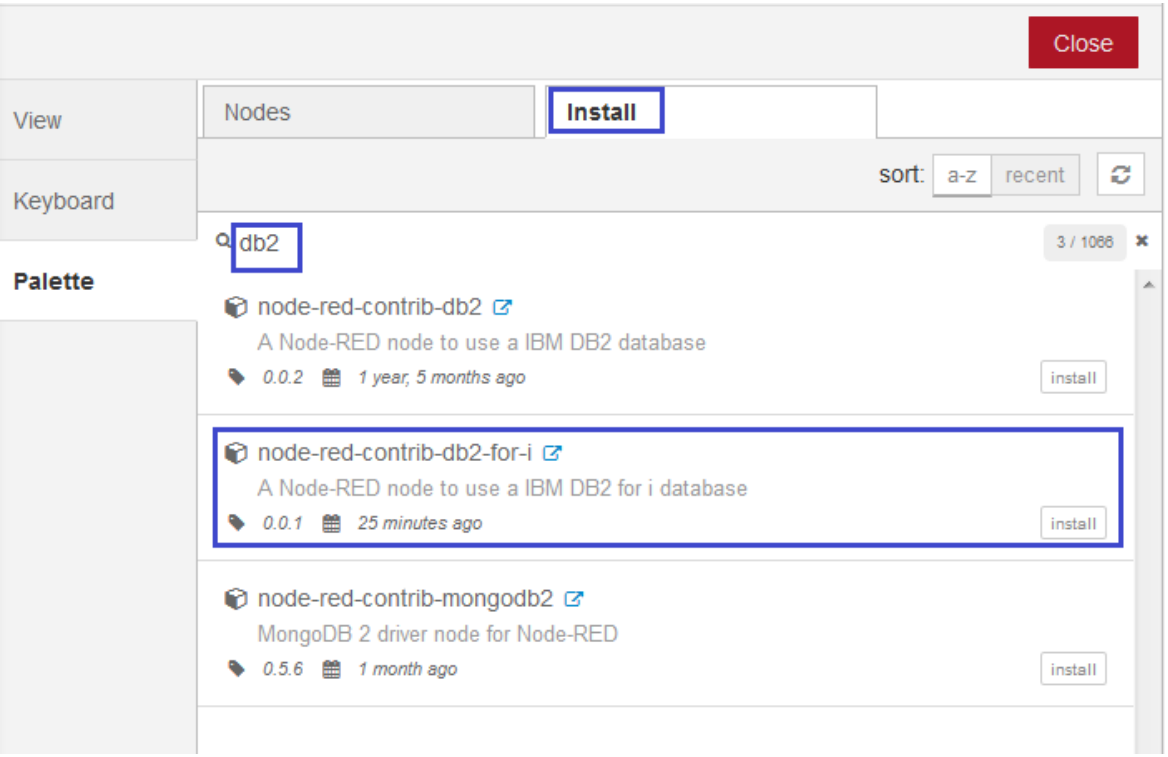
First, we need to install the Db2 for i node in our Node-RED palette. This only works on a Node-Red running on ibm i

- In Node-RED, click the button at the upper-right corner and click **Manage palette**.



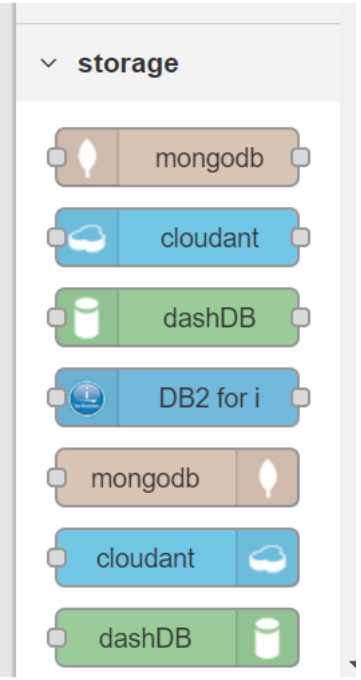
DB2 integration

- On the **Install** tab, search for **db2**, and then search for **node-red-contrib-db2-for-i**, and click **Install**. Wait for the installation confirmation dialog box to be displayed, and then restart Node-RED.



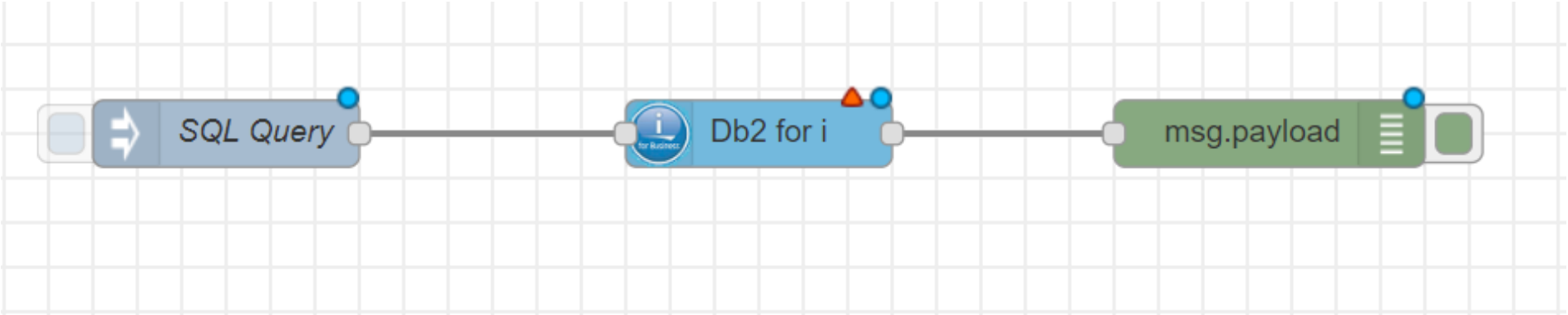
DB2 integration

- Browse the node palette on the left side and notice that the **Db2 for i** node is available in the **Storage** category.



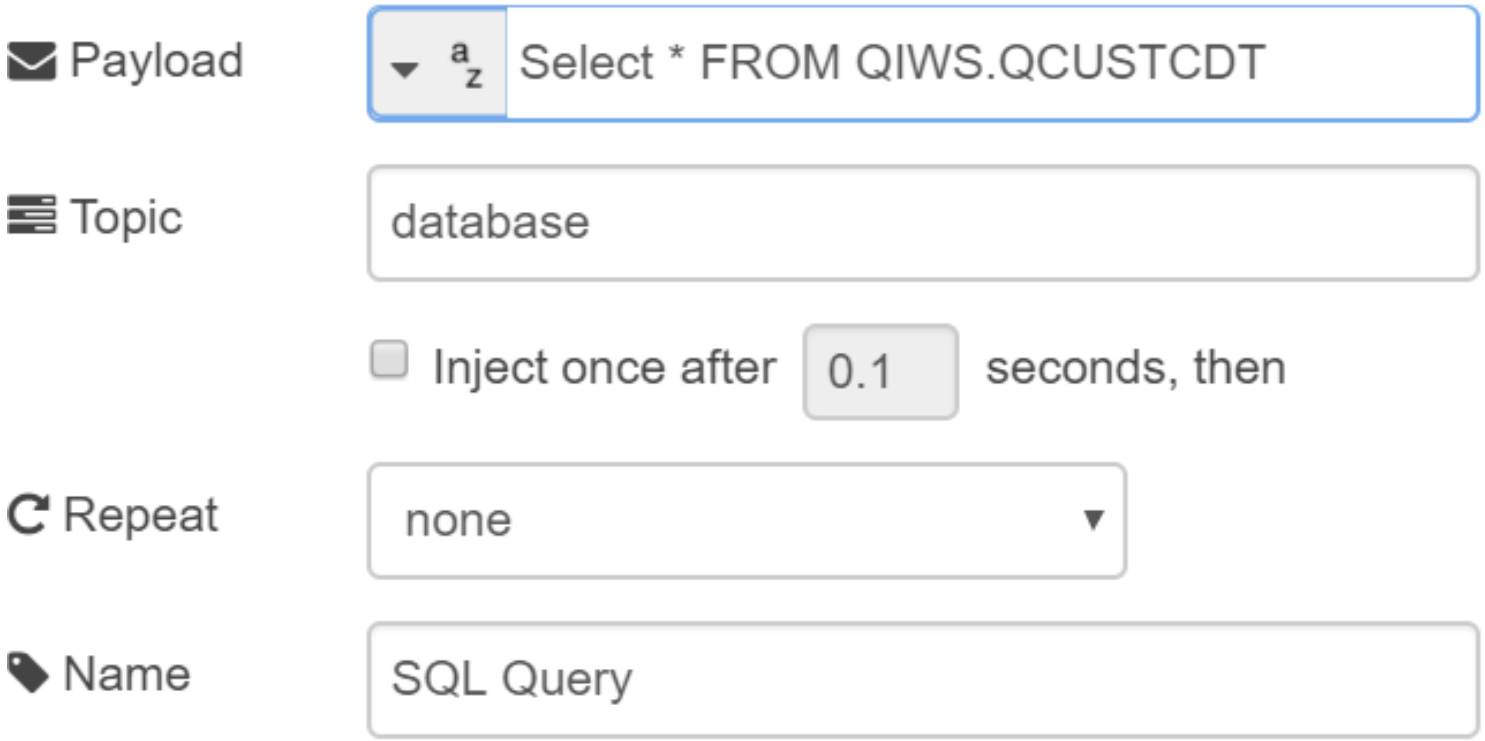
DB2 integration

Let's test the node with a simple query. Drag the **Db2 for i** node to a new flow, as shown in the following figure. Then, add **SQL Query** as the inject node and **msg.payload** as the debug node.



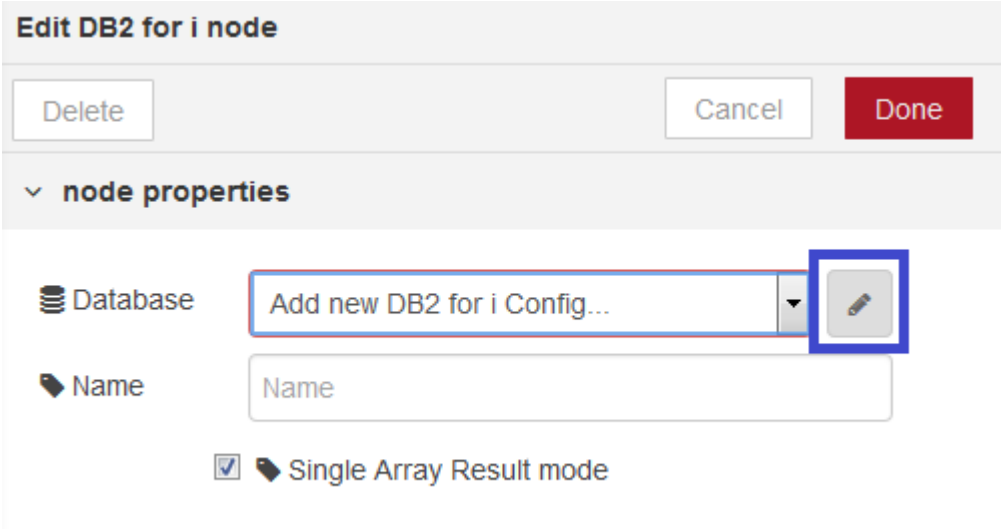
DB2 integration

- Configure the **Inject** node with the settings and SQL query as shown in the following figure.



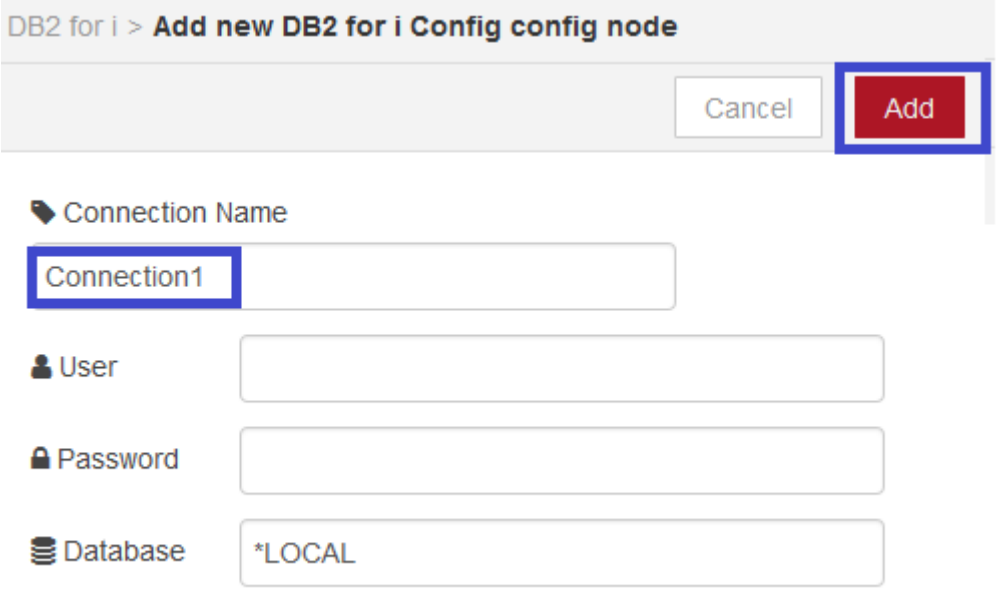
DB2 integration

- Configure the **Db2 for i** node with the settings as shown in the following figure.



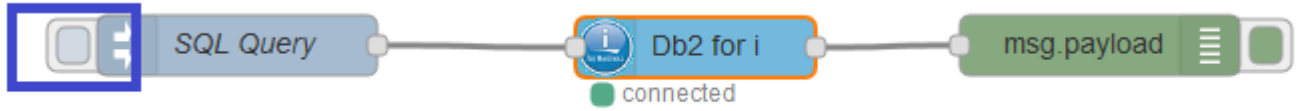
DB2 integration

- Set up a configuration node by specifying a connection name and, optionally, a user name/password if you don't want to use the current user profile running Node-RED, and you want to specify a particular user profile for connecting your database. Then click **Add** and **Done**.



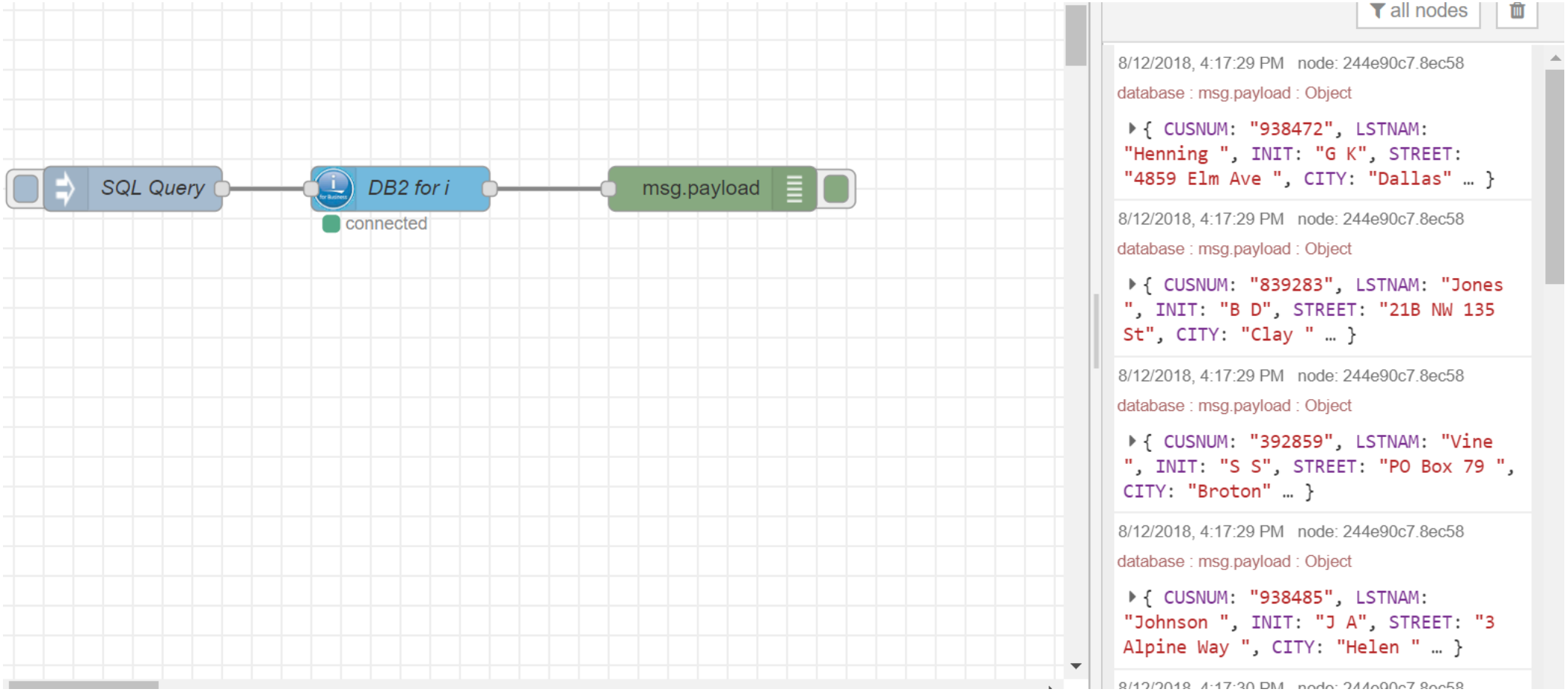
DB2 integration

- Test this simple flow by clicking **Deploy** and then click the **Inject node** button on the left side of your flow as shown in the following figure.



DB2 integration

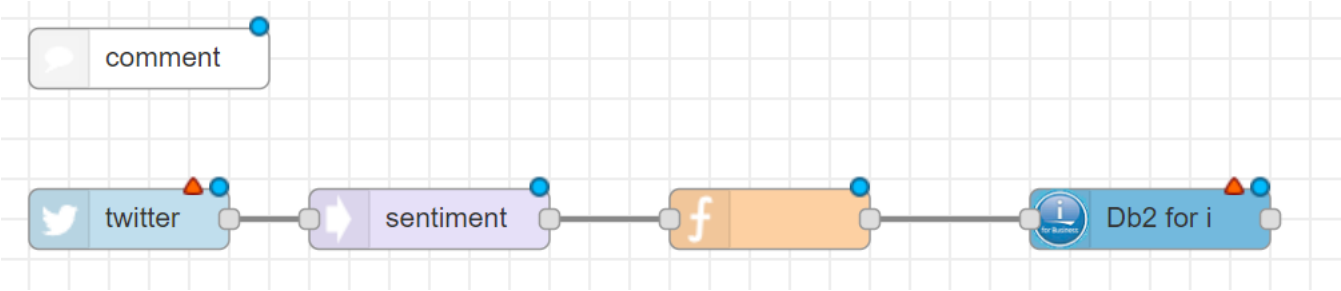
- Look into the results (JSON array of rows) of the SQL query in the **Debug** panel.



Call Twitter APIs and feed your Db2 for i database with tweets – IBM example (see links)

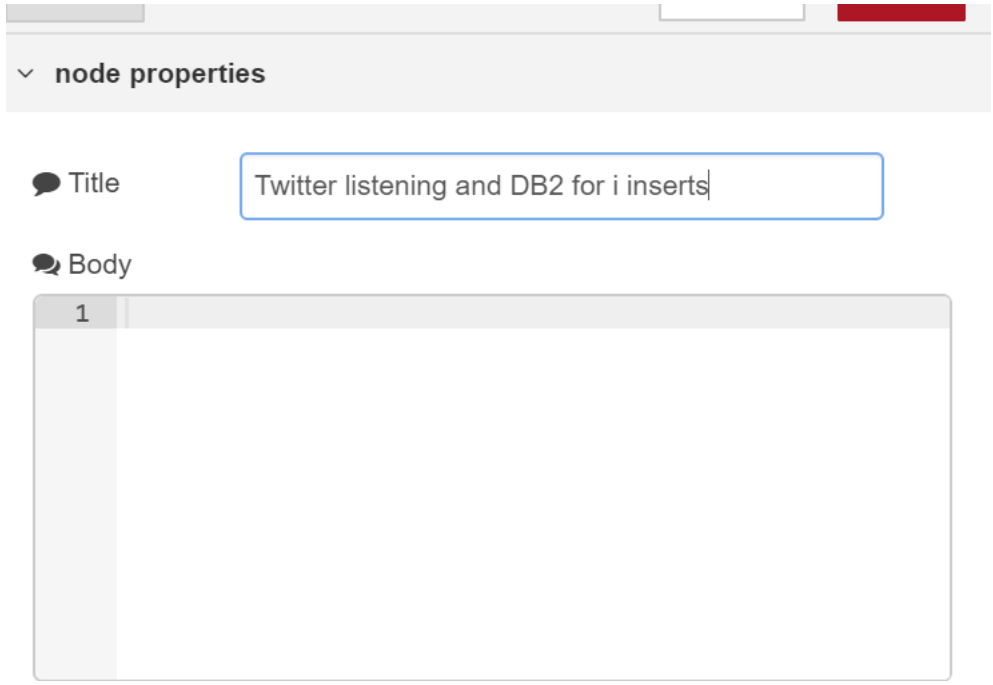
- To do so, you just have to manually drag and configure five nodes in your Node-RED flow editor.

- A comment node
- A twitter input node
- A sentiment analysis node
- A function node
- A DB2 for i node



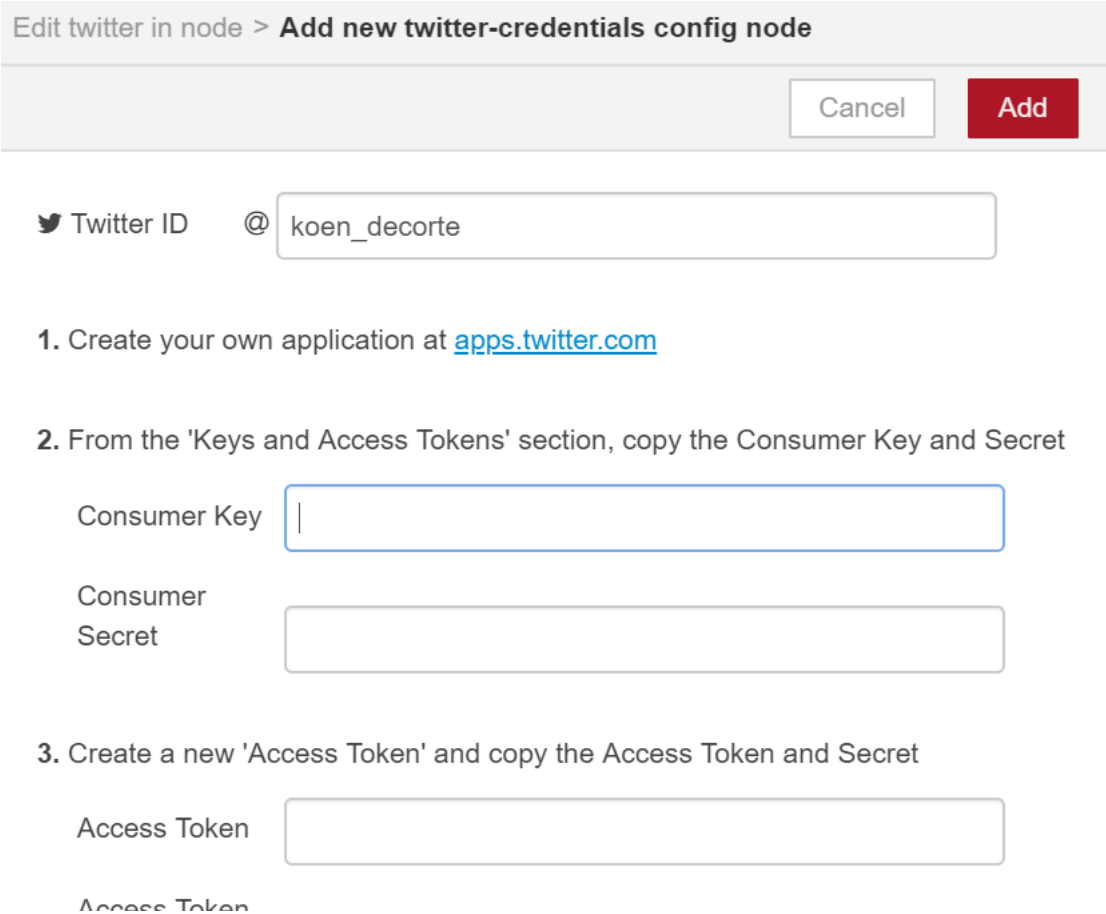
DB2 integration

Change the comment node as follows :



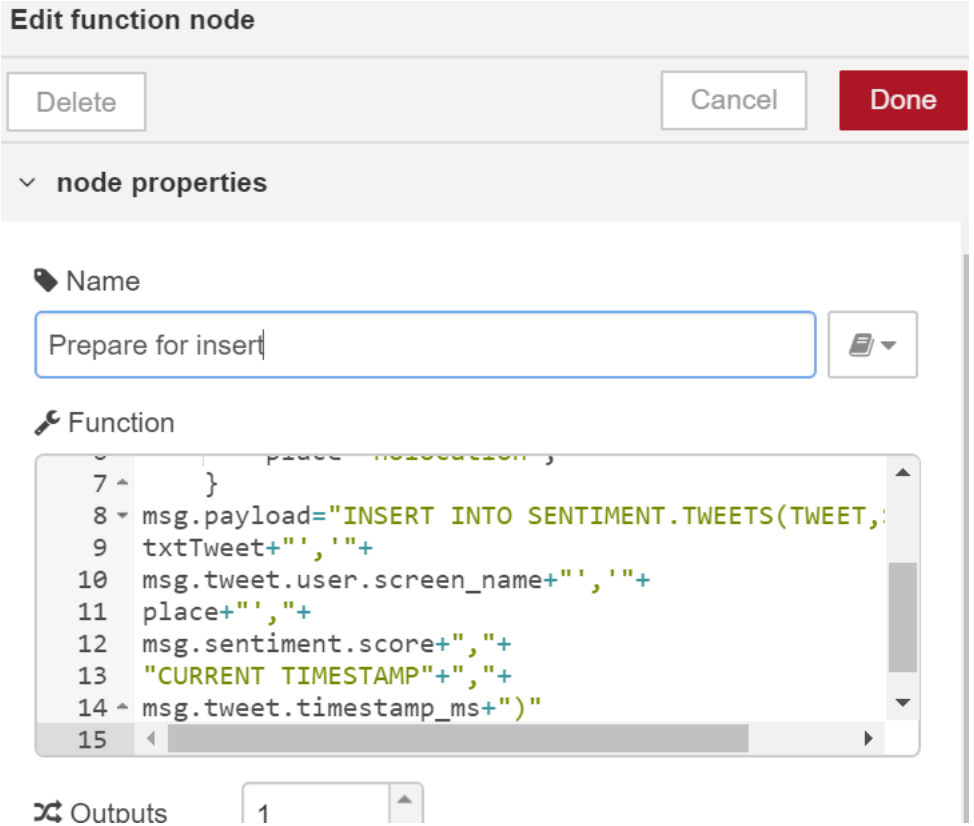
DB2 integration

Configure the twitter node :



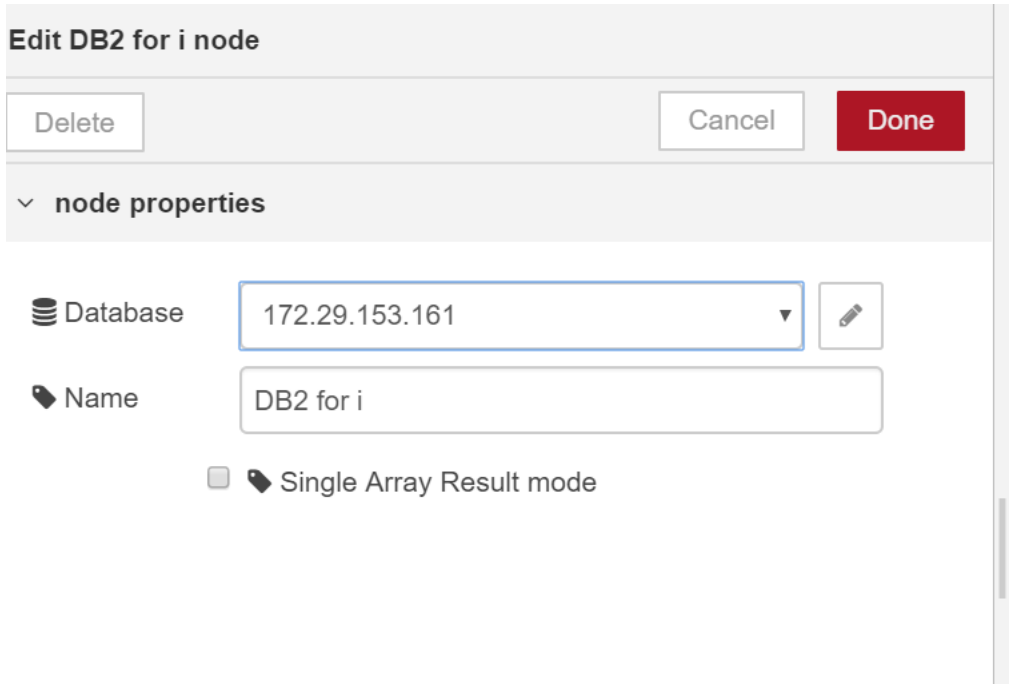
DB2 integration

Copy the function javascript in the function node and name it prepare for insert



DB2 integration

Edit the DB2 for i connection



DB2 integration

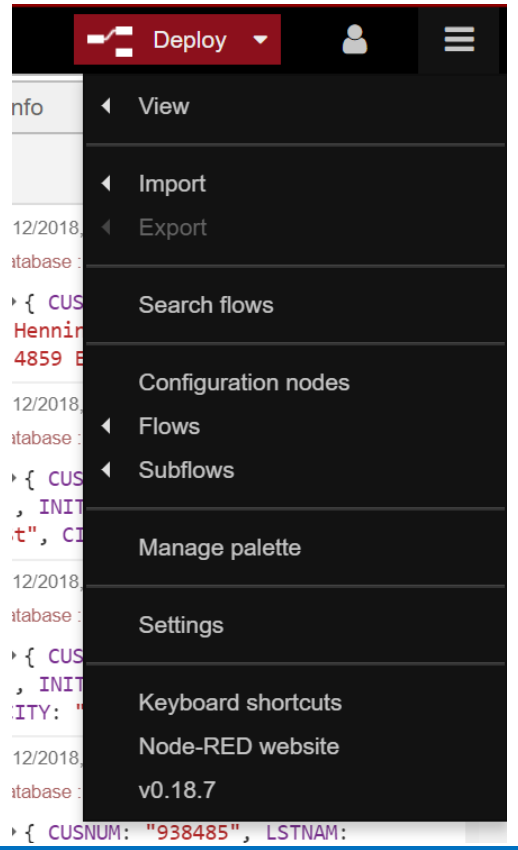
- Click **Deploy** and wait for Twitter activity. You can optionally add a **Debug** node in your flow to see the Twitter activity in the Debug view.
- Verify that your Tweets table is being populated by querying your table occasionally, either from your favorite SQL editor or from a Node-RED flow.

DB2 integration

We can enrich incoming tweets with IBM Watson services, for example, to translate the incoming tweets into English on the fly before performing sentiment analysis.

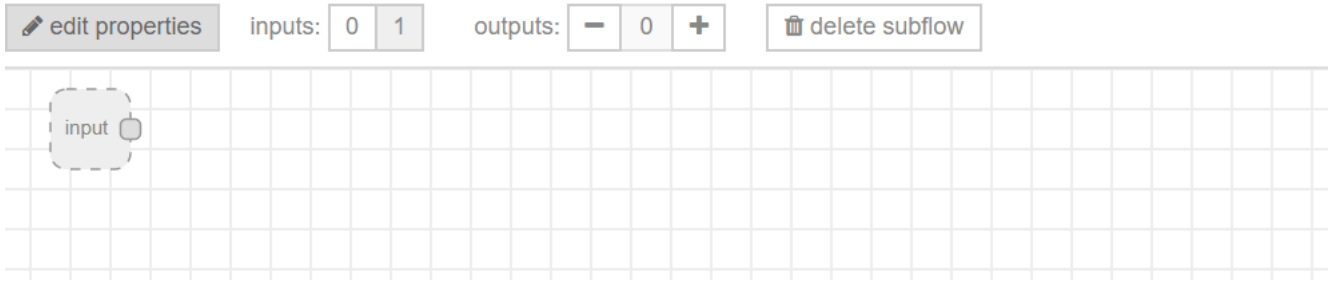
First install **node-red-node-watson** locally.

Create a subflow

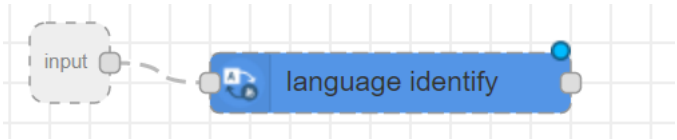


DB2 integration

- Click 1 on input to add an input to the subflow.

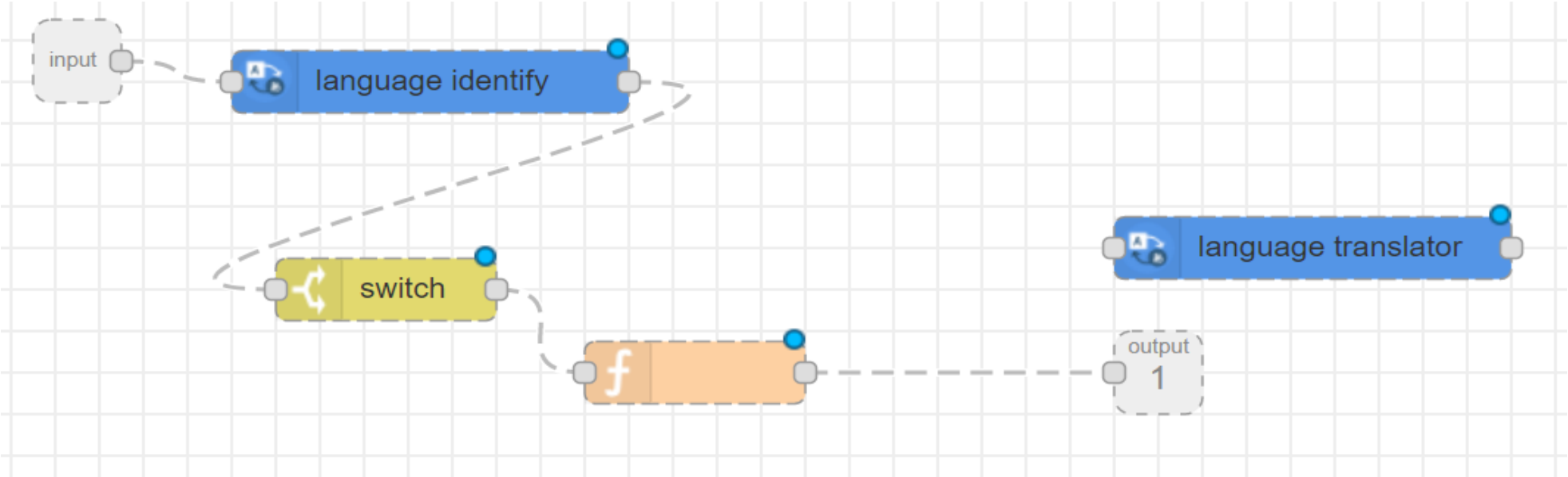


- Add language identify node and link it to the input node



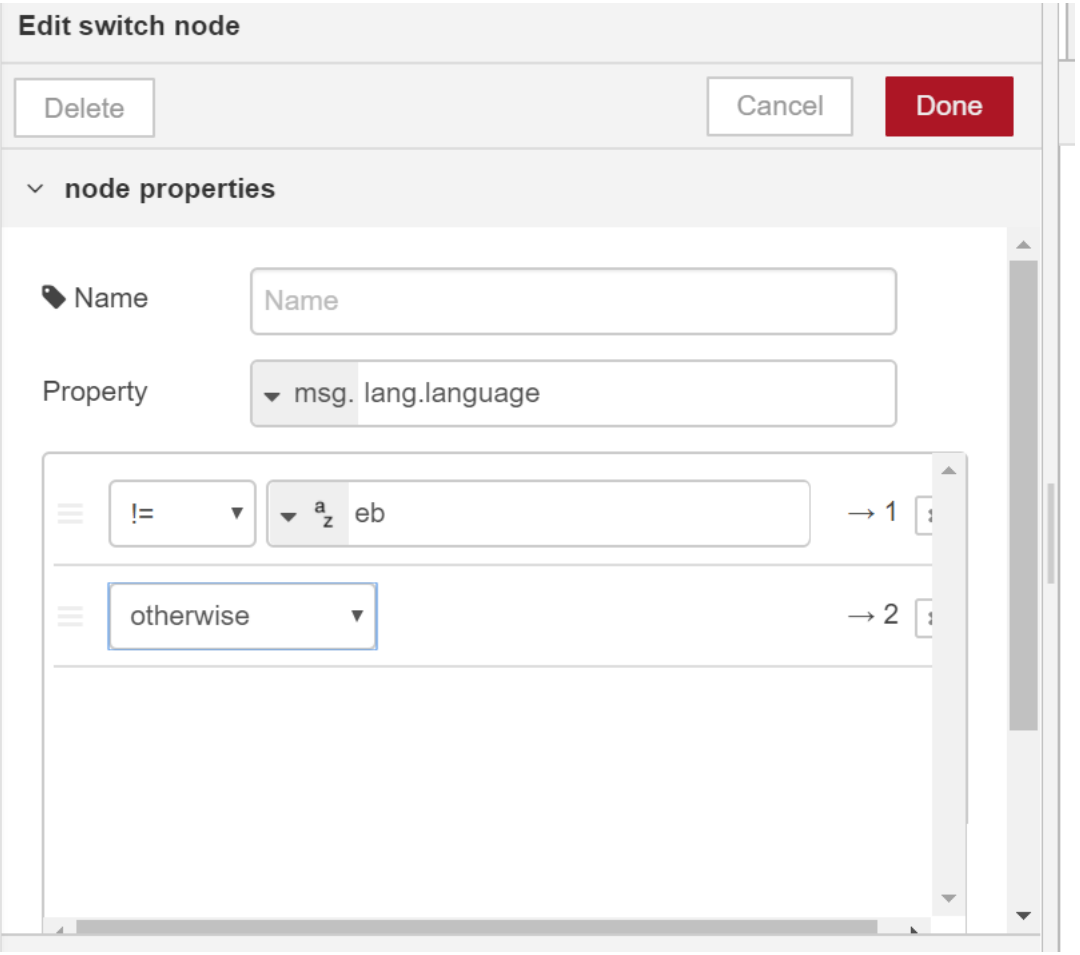
DB2 integration

- Add a Watson translator node and configure the two Watson nodes (by specifying the user name, password, and endpoint if needed), each pointing to the same IBM Watson service you have just instantiated on IBM Cloud, but each using a different API function (language identity and language translation).
- Add a switch node and link it to the language identify, add a function and link it to the switch add an output and link it to the function



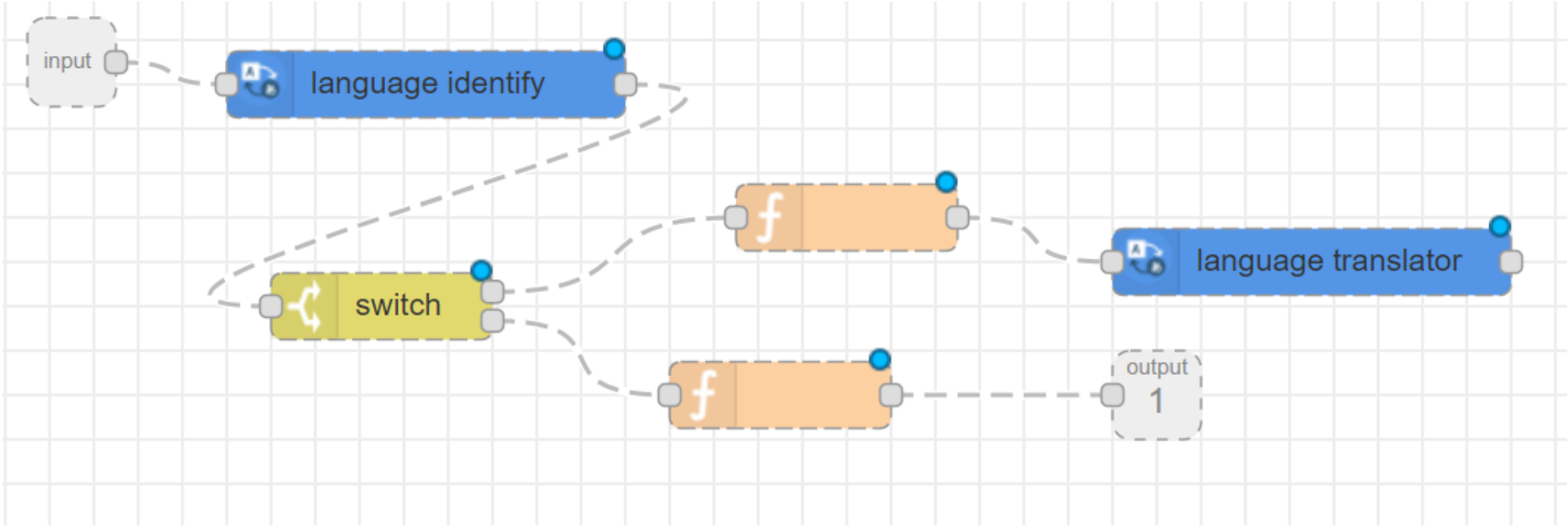
DB2 integration

- Edit the switch node as follows :



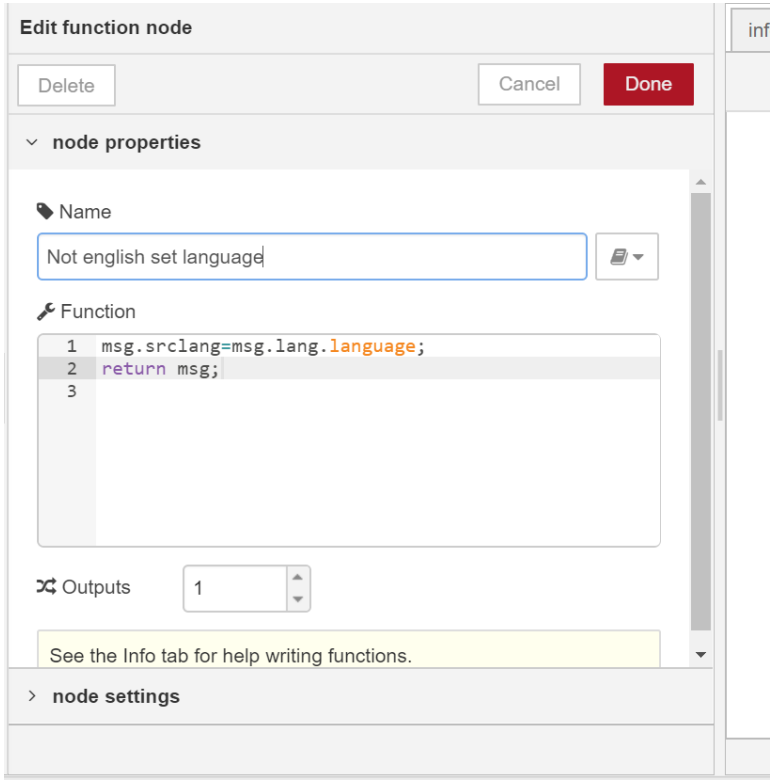
DB2 integration

- Add a new function and link it to the top switch and language translator.
- Link translator to the bottom function
- Link the bottom switch to the function linked with the output



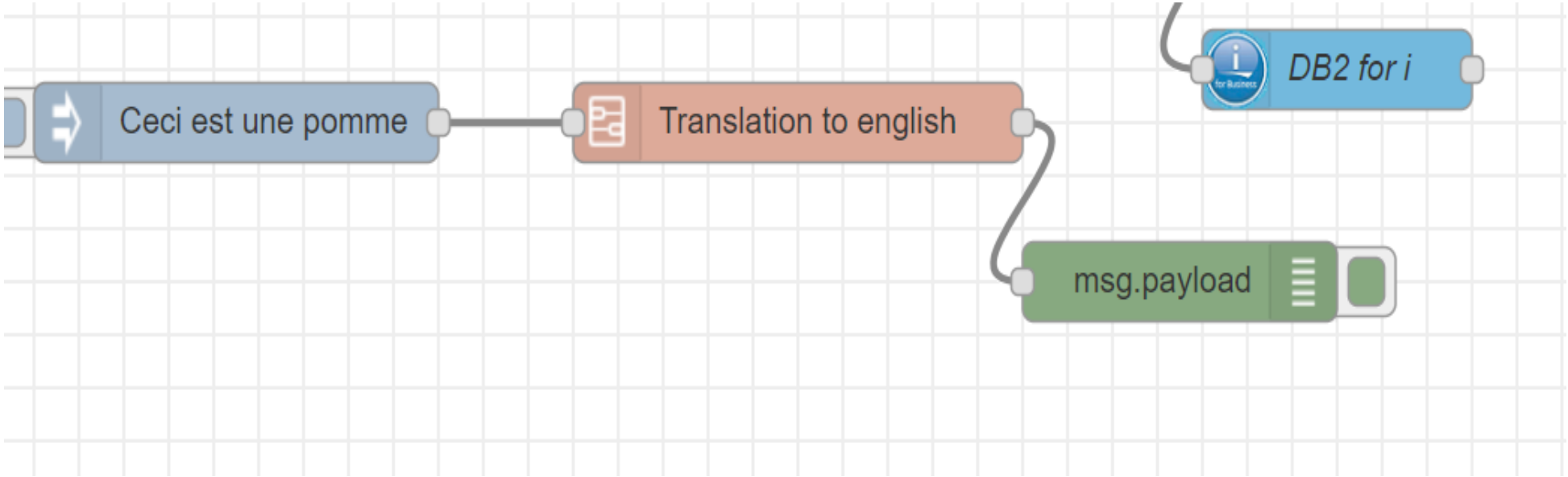
DB2 integration

- Edit the function before the translate node as follows :



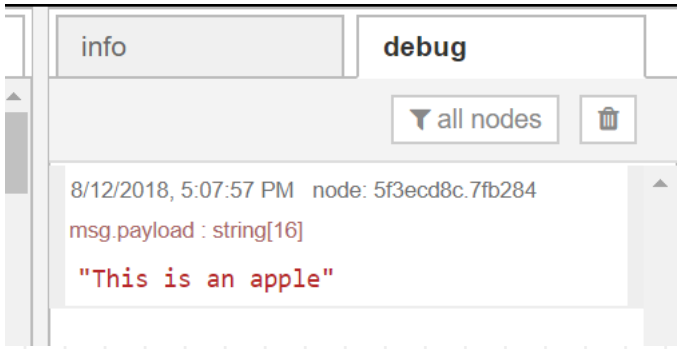
DB2 integration

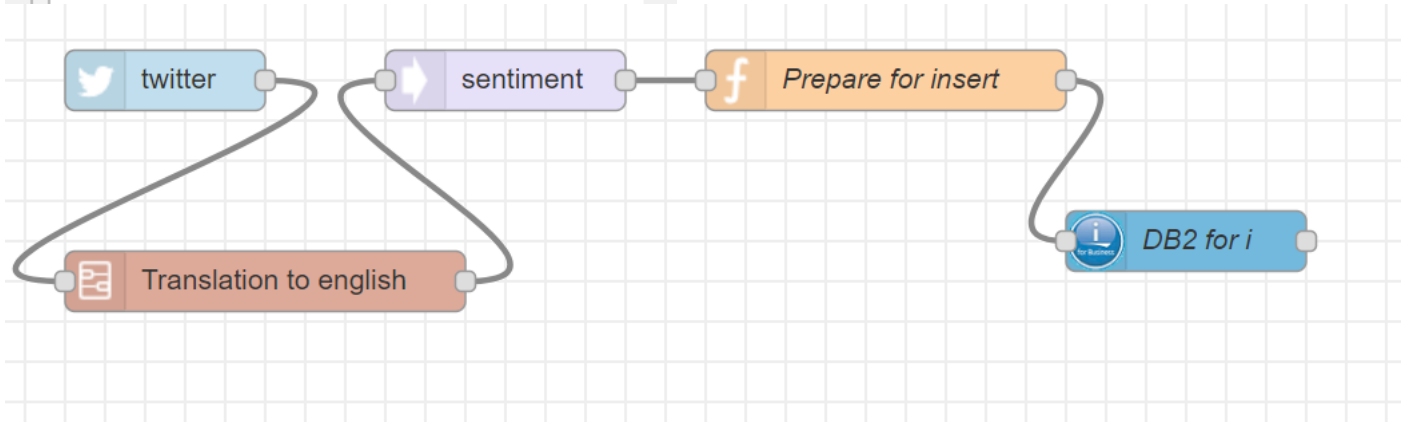
- Deploy the subflow. Add it to the twitter insert flow and add an input and debug node to it.



DB2 integration

- Change the input node to string and write a string in French. Deploy and click to view the debug data. Then delete the input node and debug node and insert the translation subflow before the sentiment node



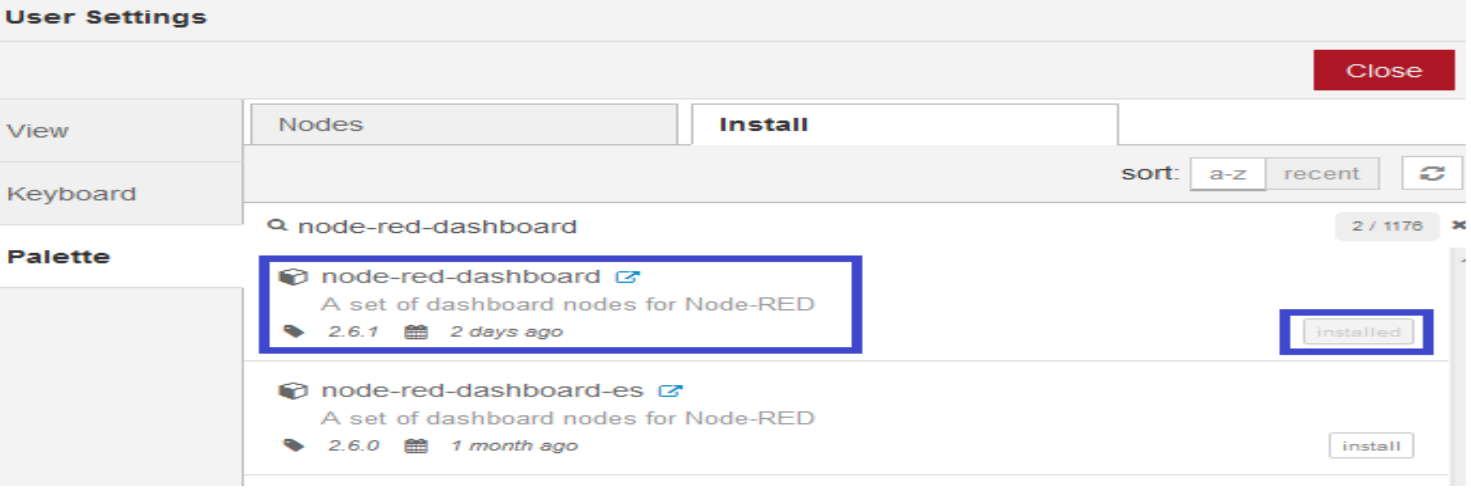


DB2 integration

- Create a social media dashboard, mixing your business data on Db2 for i with data coming from the IBM Cloud and IoT.

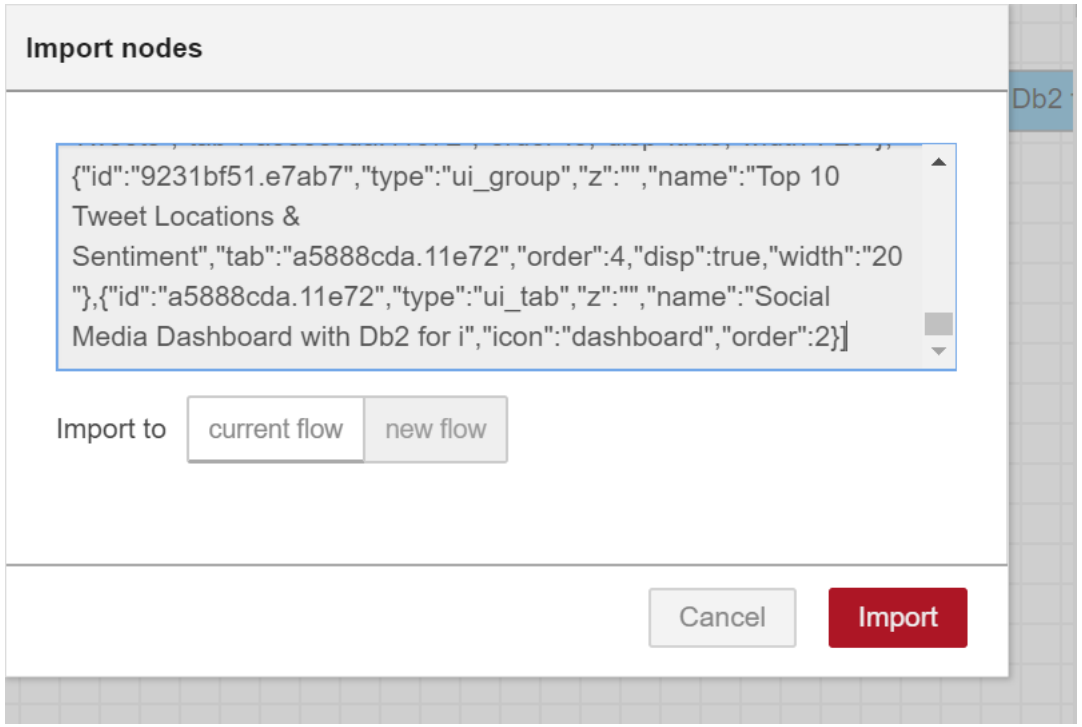
Install the dashboard nodes.

Many Node-RED exist for generating UI and graphs.
From your Node-RED editor, in the upper-right menu, click **Manage palette**, and search for **node-red-dashboard**. Then, click **Install** and restart Node-RED.

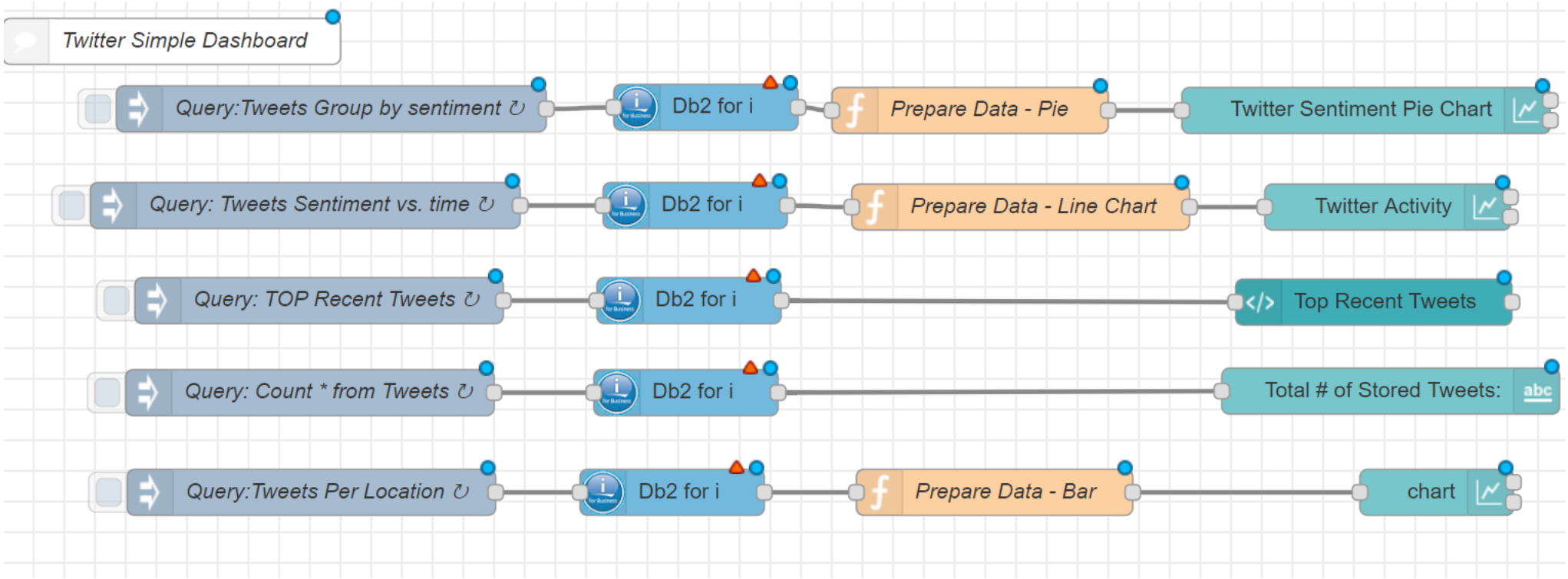


DB2 integration

- Import the JSON file as a flow and change your IBM i settings. The database is provided with the SQL file.

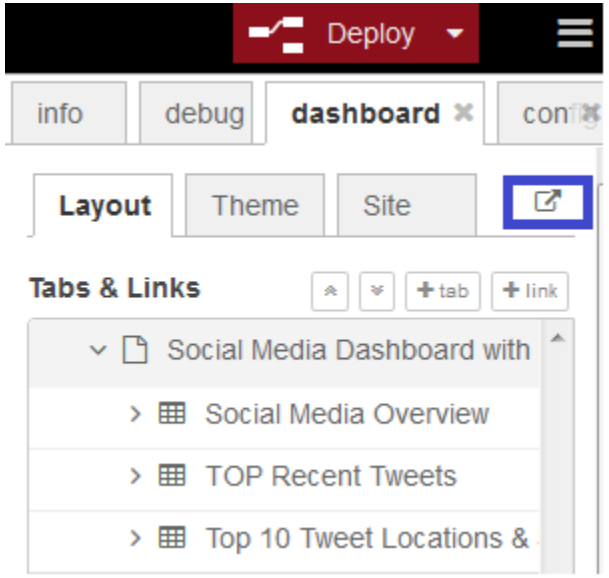


DB2 integration



DB2 integration

- Change the db2 nodes to your system settings.
- Deploy and access your dashboard. The refresh rate in the just imported code is set to 10 minutes. So, if you don't want to wait, go to your flow editor and force a dashboard refresh by clicking each inject nodes on the left side of your flow. The dashboard URL is available at the right side of the dashboard panel.

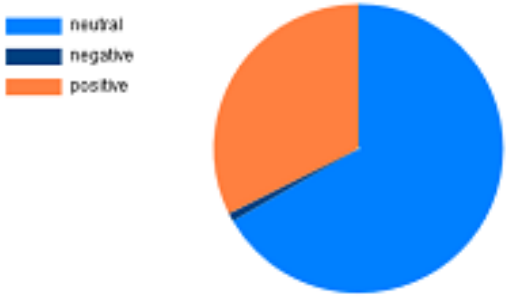


DB2 integration

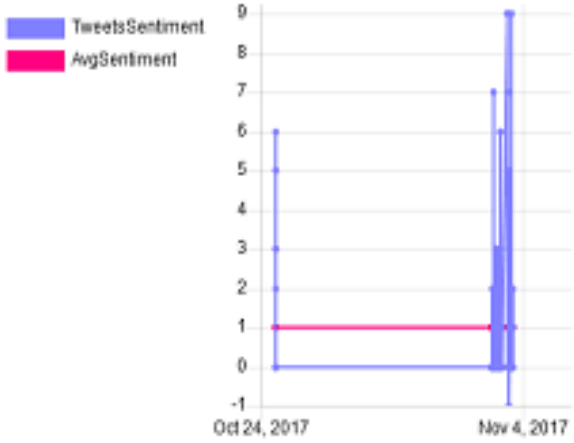
Social Media Dashboard with Db2 for i

Social Media Overview

Twitter Sentiment for #IBMi



Twitter Activity

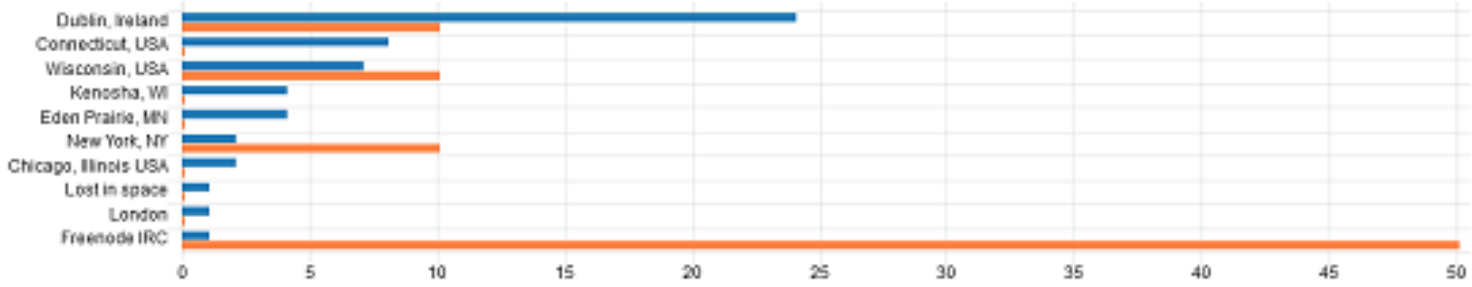


TOP Recent Tweets

Tweet	Name	Location	Sentiment	TimeStamp
RT: @FloydDelMuro-See how walls co	michentr	Dublin, Ireland	0	2017-11-03-14.55.36.464683
See how walls continue to crumble	FloydDelMuro	Chicago IL	0	2017-11-03-14.55.36.321790
The latest PHP on #IBMi #IN! https	gmantechi	Delavan WI	2	2017-11-03-14.17.52.764318
RT: @COMMONug-RT @J_Buck51: #acces	michentr	Dublin, Ireland	1	2017-11-03-14.17.08.795327
RT @J_Buck51: #accessclientsolutio	COMMONug	Chicago, Illinois USA	1	2017-11-03-14.17.08.533199
RT @COMMONug: Get to know #AccessC	jbuck_imPower	Wisconsin, USA	0	2017-11-03-13.47.40.825775
RT @AndyYouens: Thank you all who	jbuck_imPower	Wisconsin, USA	9	2017-11-03-13.47.32.009431
RT @petem59: Where is option to se	jbuck_imPower	Wisconsin, USA	0	2017-11-03-13.46.52.783031

Total # of Stored Tweets: **108**

Top 10 Tweet Locations & Sentiment



Node-RED

Deploy

filter nodes

Flow 1

input

inject
catch
status
link
mqtt
http
websocket
tcp
udp
s7
s7 http get
s7comm

output

debug
link
mqtt
http response
websocket
tcp
udp
s7
s7comm

function

Monitor variables
connecting
all variables
[get] /setn okhpl
set stop hpl afmeting ok
[get] /setokhpl
set hpl afmeting ok
afmetingen HPL
connecting
Check Lijm
[get] /setAantalPlaten
Aantal Platen
Aantal_platen
connecting
Uitvoer Lijmstraat monitor
connecting
Uitvoer variables
Pos 104
Doorschuiven
[get] /startmachine
set start
Start
connecting
[get] /stopmachine
set stop
[get] /setdragerb
breedte
Com_xGet_Plaat-Breedte
connecting
lengte
Com_xGet_Plaat-Lengte
connecting
[get] /setdragerl
catch all output
[get] /setdragerd
dikte
Com_xGet_Plaat-Dikte
connecting
[get] /sethplb
breedte
Com_xGet_HPL_Breedte
connecting
[get] /sethpll
lengte
Com_xGet_HPL_Lengte
connecting
[get] /sethpld
dikte
Com_xGet_HPL-Dikte
connecting
[get] /setkarton
karton
karton
connecting
delay 5s
karton

info

Information

Flow	"e16a9fea.afb6b"
Name	Flow 1
Status	Enabled

Flow Description

None

Enable or disable these tips from the option in the settings

Node-RED

Deploy

filter nodes

Flow 1

input

inject
catch
status
link
mqtt
http
websocket
tcp
udp
s7
s7 http get
s7comm

output

debug
link
mqtt
http response
websocket
tcp
udp
s7
s7comm

function

[get] /SetTemp
Temperatuur
Temperatuur
online
[get] /SetDruk
Druk
Druk
online
msg
[get] /SetTijd
Tijd
Tijd
online
[get] /SetDikte
Dikte
Dikte
online
[get] /StartPers
Start
Start
online
delay 1s
Start
[get] /UnloadPers
Unload
Unload
online
delay 1s
Unload
[get] /SetPistons
Piston1
Piston1
online
Piston2
Piston2
online
Piston3
Piston3
online
Piston8
Piston4
Piston4
online
Piston9
Piston5
Piston5
online
Piston10
Piston6
Piston6
online
Piston11
Piston7
Piston7
online
Piston12
Druk_inschakel
Piston8
online
Piston9
online
Piston10
online
Piston11
online
Piston12
online
Druk_Inschakel
online
[get] /setDrukSet
Druk_set
Druk_Set
online
[get] /SetAfstapel
Teller
Unload
connecting
delay 1s
Teller

info

Information

Flow	"e16a9fea.afb6b"
Name	Flow 1
Status	Enabled

Flow Description

None

Dragging a node onto a wire will splice it into the link

CRAS WOODSHOP

WORKCENTER STATUS

TEMPERATURE 80

HUMIDITY 20

RUNNING NORMALLY

0/0 TODAY 01.11.2018

0/0 CURRENT BATCH

0/0 CURRENT BUCKET

0/0 PICKING 01.11.2018

0 PICKINGERRORS

Druk op F11 om het volledige scherm te sluiten

Batches overview

New batch | Planning | Refresh | Settings | Sequence

Batch ...	Seque...	Description	Production...	Number todo	Status description	Picking HPL to cut	Cutting HPL	Picking HPL	Picking plates to cut	Cutting plates	Picking plates	Transport	Machine
56	20	Heule 20/09	20/09/2018	70	Pick instructies aang...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
58	10	Heule 27.09	27/09/2018	10	Pick instructies aang...	DONE	DONE	DONE	DONE	DONE	DONE	DONE	TO DO
59	20	Tes	27/09/2018	8	Gepland	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
62	30	TEST 01	27/09/2018	15	Eerste lijn gepickt / P...	DONE	DONE	DONE	DONE	BUSY	TO DO	TO DO	TO DO
66	10	qsfqfq	03/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
67	20	TEST	03/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
65	10	310 versie 2	05/10/2018	36	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	DONE	DONE	TO DO
68	10	5 / 10	08/10/2018	36	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	BUSY	TO DO	TO DO
69	10	10 10 18 order 1	10/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	DONE	DONE	BUSY
70	20	10 10 18 order 2	10/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	DONE	DONE	BUSY
71	30	j	10/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
72	40	10 10 18 ORDER 3	10/10/2018	3	Pick instructies aang...	DONE	DONE	DONE	NOT NEEDED	NOT NEEDED	DONE	DONE	BUSY
74	50	10 10 18 ORDER 4	10/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	DONE	DONE	BUSY
75	60	10 10 18 order 5	10/10/2018	0	Pickprocess gestart	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
77	80	10 10 18 ORDER 5 BIS	10/10/2018	1	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	DONE	DONE	DONE	DONE	BUSY
78	10	TEST MACHINE	12/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
79	20	TEST MACHINE	12/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
80	30	TEST MACHINE	12/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
81	40	TEST MACHINE	12/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
171	10	test scenario 1	29/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	BUSY	TO DO	TO DO
172	20	scen 1	29/10/2018	0	Gepland	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO
175	40	scenario 2	29/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	DONE	NOT NEEDED	NOT NEEDED	BUSY	TO DO	TO DO
178	60	scenario 3	29/10/2018	2	Pick instructies aang...	BUSY	TO DO	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
179	70	scenario 4	29/10/2018	2	Pick instructies aang...	NOT NEEDED	NOT NEEDED	BUSY	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
180	80	scenario 5	29/10/2018	0	Pickprocess gestart	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO	TO DO

CRAS WOODSHOP

WORKCENTER STATUS

TEMPERATURE 80

HUMIDITY 20

RUNNING NORMALLY

0/0	0/0	0/0	0/0	0 PICKINGERRORS
TODAY 01.11.2018	CURRENT BATCH	CURRENT BUCKET	PICKING 01.11.2018	

Batchid: 56

Description: Heule 20/09

Status: Pick instructies aangemaakt

Number todo: 70

End date: 20/09/2018

Number done: 0

Batches | Planningview | Stock problems | Delete batch | Reopen batch | Start picking | Picking list | Pickinstructions list | Cutting list | Transport list | Machine list

Buckets 56 Heule 20/09

Click to collapse. CTRL/click collapses all others

Bucketid	Produc...	Customer	Temperatu...	Time	Pressure	Quantity	Unit	Status	Picking HPL to cut	Cutting HPL	Picking HPL	Picking plates to cut	Cutting plates	Picking plates	Transport	Machine
Bucketid: 1 HPL02/MDF01																
1 HPL0...	150004...	K00456	11	0	0	2	ST			BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
1 HPL0...	150004...	K26278	11	0	0	2	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
1 HPL0...	150004...	K00613	11	0	0	1	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
1 HPL0...	150004...	K00515	11	0	0	1	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
1 HPL0...	150004...	K00484	11	0	0	1	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
1 HPL0...	150004...	K02082	11	0	0	2	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
	6					9										
Bucketid: 2 HPL01/MTX01																
2 HPL0...	150003...	K00823	12	45	15	50	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
	1					50										
Bucketid: 3 HPL01/MTX01																
3 HPL0...	150003...	K00823	12	45	15	2	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
	1					2										
Bucketid: 4 HPL02/MTX01																
4 HPL0...	150004...	K05462	17	0	0	2	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
	1					2										
Bucketid: 5 HPL03/MDF02																
5 HPL0...	150003...	K00061	24	0	0	7	ST	Pick instructies ...	DONE	BUSY	TO DO	NOT NEEDED	NOT NEEDED	TO DO	TO DO	TO DO
	1					7										

Batches to start | Machine overview | Machine operations

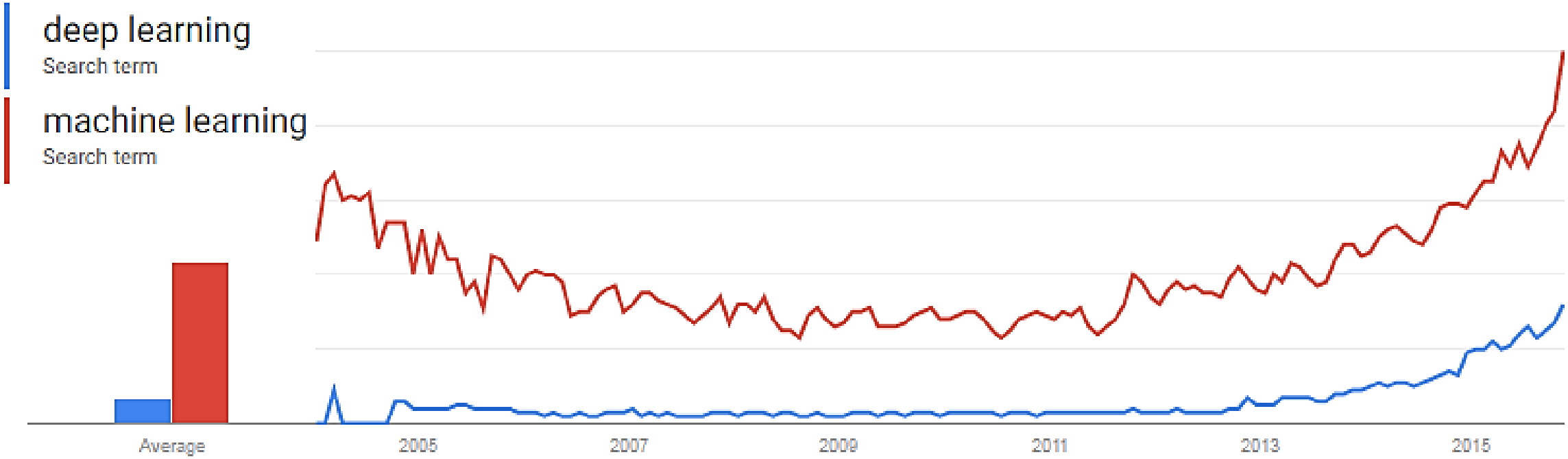
Start
Stop
Test batch
Set temperature
Set time
Set pressure
Set karton
Doorschuiven
Reset teller
Unload pers
End order
Calibration
Reset order
Reset afstapelaar


ORMA
MACCHINE
ORMA
MACCHINE

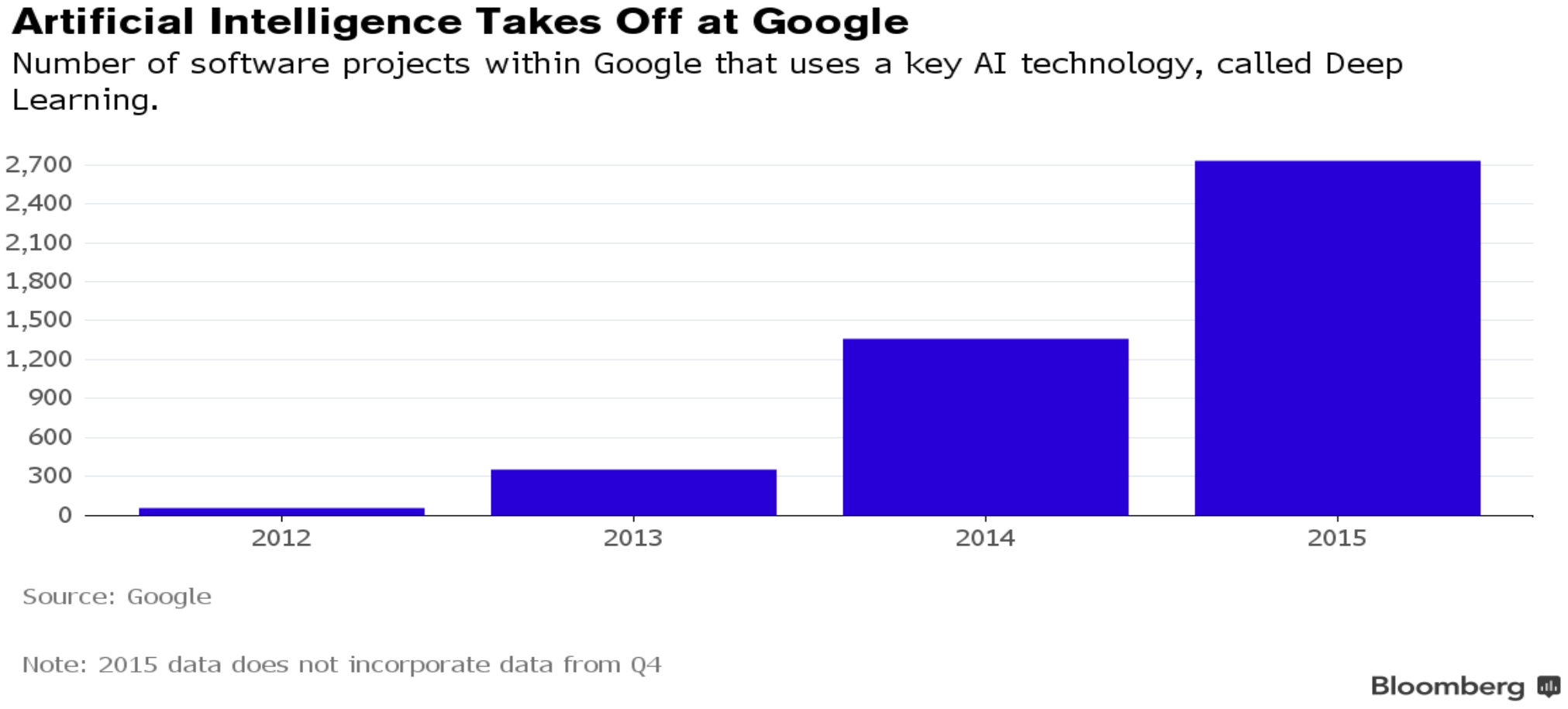

ORMA

DEEP LEARNING

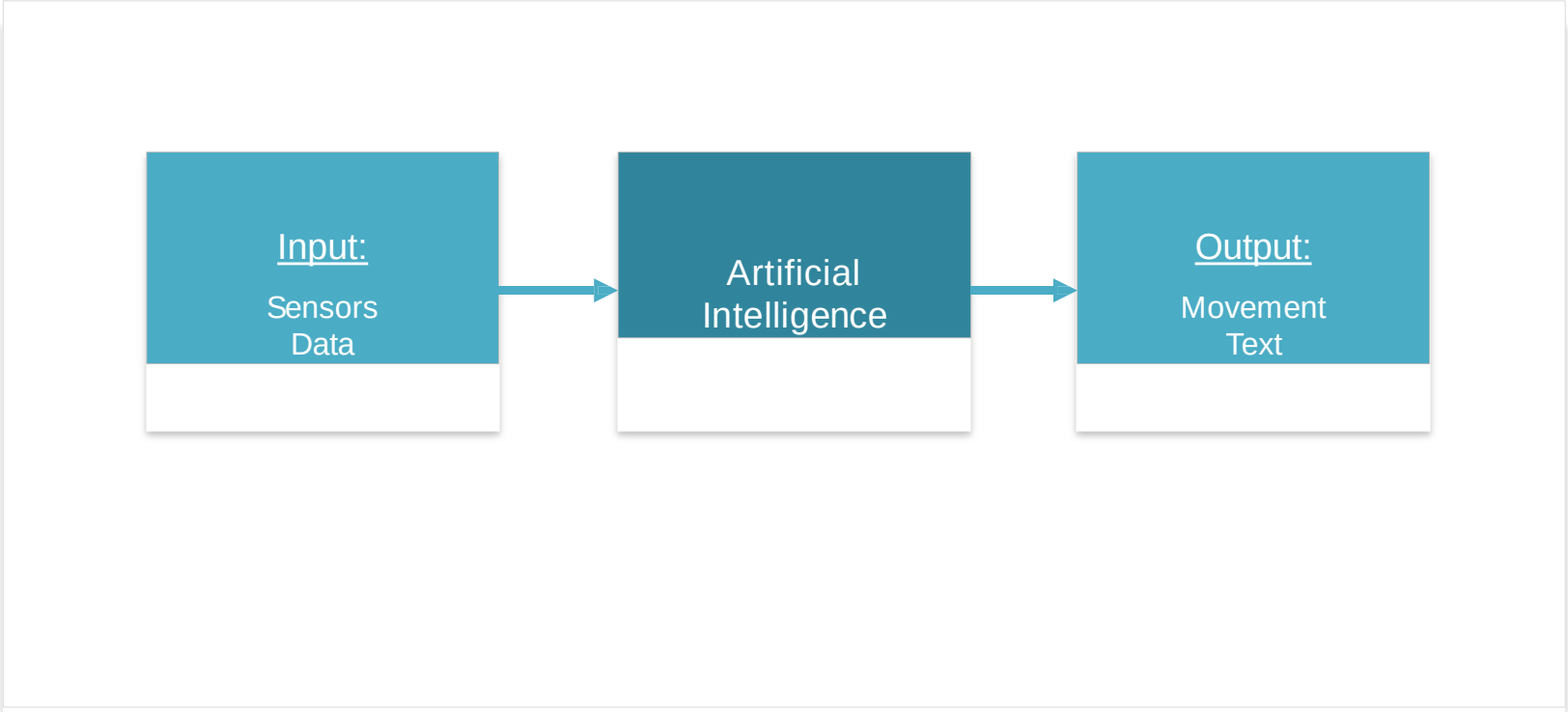
Google search terms



Hype or reality ?

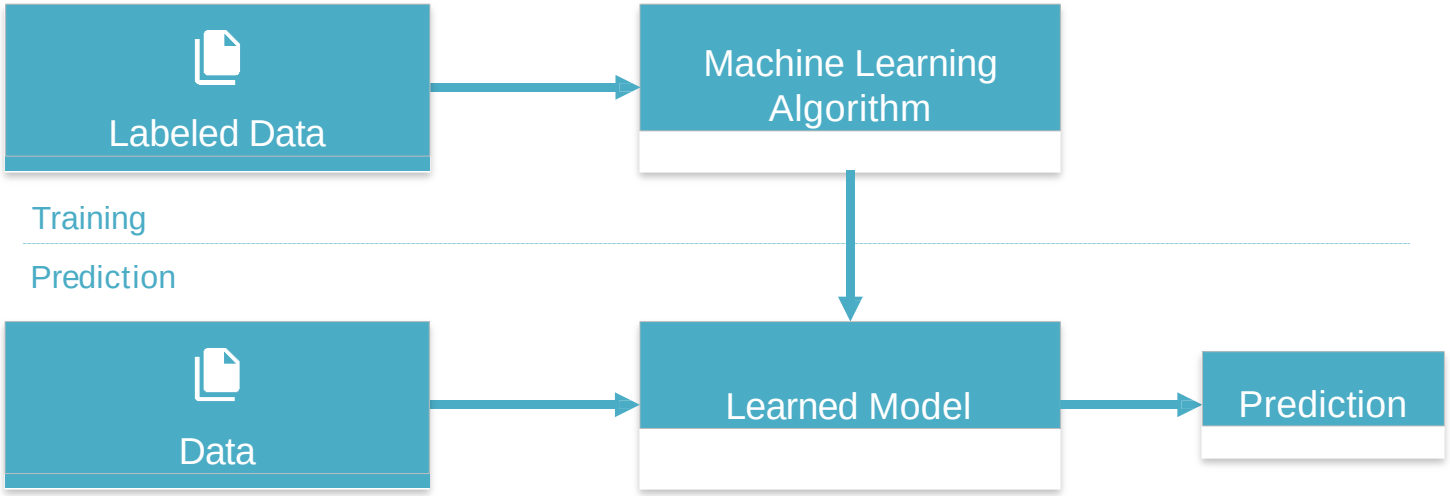


What is Artificial Intelligence?



Machine Learning - Basics

Machine Learning is a type of Artificial Intelligence that provides computers with the ability to learn without being explicitly programmed.



Provides various techniques that can learn from and make predictions on data

Machine Learning - Learning Approaches

Supervised Learning: Learning with a labeled training set
Example: email spam detector with training set of already labeled emails

Unsupervised Learning: Discovering patterns in unlabeled data
Example: cluster similar documents based on the text content

Reinforcement Learning: learning based on feedback or reward
Example: learn to play chess by winning or losing

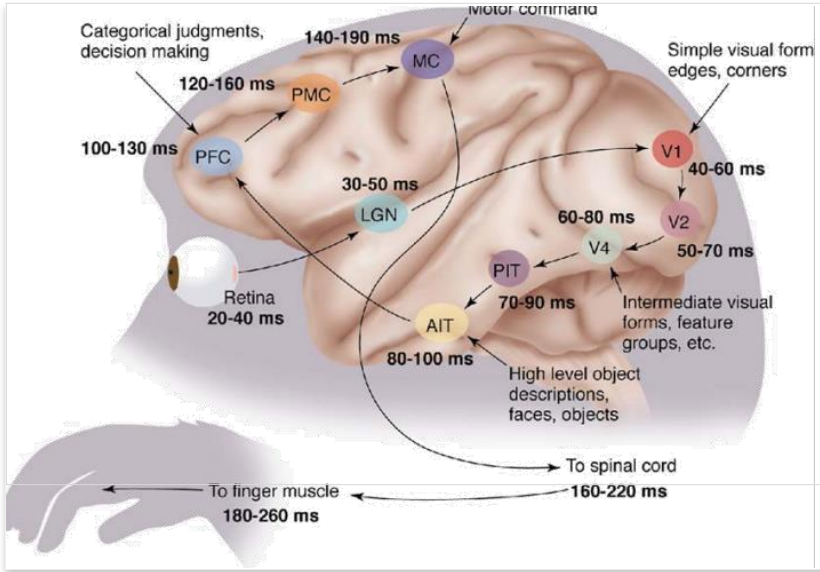
What is Deep Learning?

Part of the machine learning field of learning representations of data. Exceptional effective at learning patterns.

Utilizes learning algorithms that derive meaning out of data by using a hierarchy of multiple layers that mimic the neural networks of our brain.

If you provide the system tons of information, it begins to understand it and respond in useful ways.

Inspired by the Brain



The first hierarchy of neurons that receives information in the visual cortex are sensitive to specific edges while brainregions further down the visual pipeline are sensitive to more complex structures such asfaces.

Our brain has lots of neurons connected together and the strength of the connections between neurons represents long term knowledge.

1

One learning algorithm hypothesis: all significant mental algorithms are learned except for the learning and reward machinery itself.

Why Deep Learning?

Speech Recognition

Computer Vision

Natural Language Processing

A brief History

1958 Perceptron

1974 Backpropagation

Convolution Neural Networks for Handwritten Recognition

1998

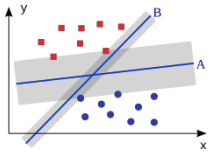
awkward silence (AI Winter)

Google Brain Project on 16k Cores

2012

1969
Perceptron criticized

1995
SVM reigns

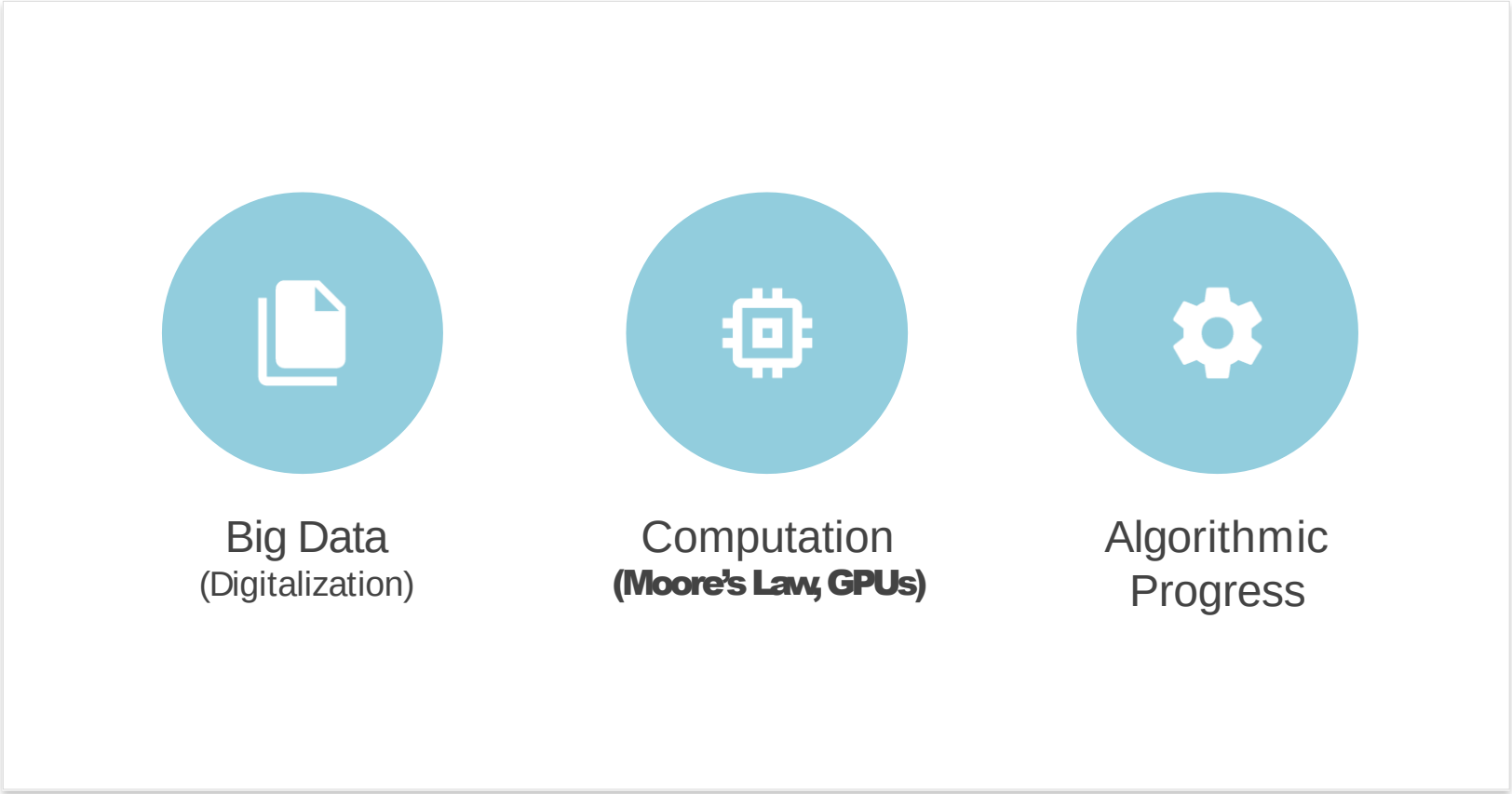


2006
Restricted Boltzmann Machine

2012
AlexNet wins ImageNet

IMAGENET

What changed? Old wine in new bottles



The Big Players

Geoffrey Hinton: University of Toronto & Google

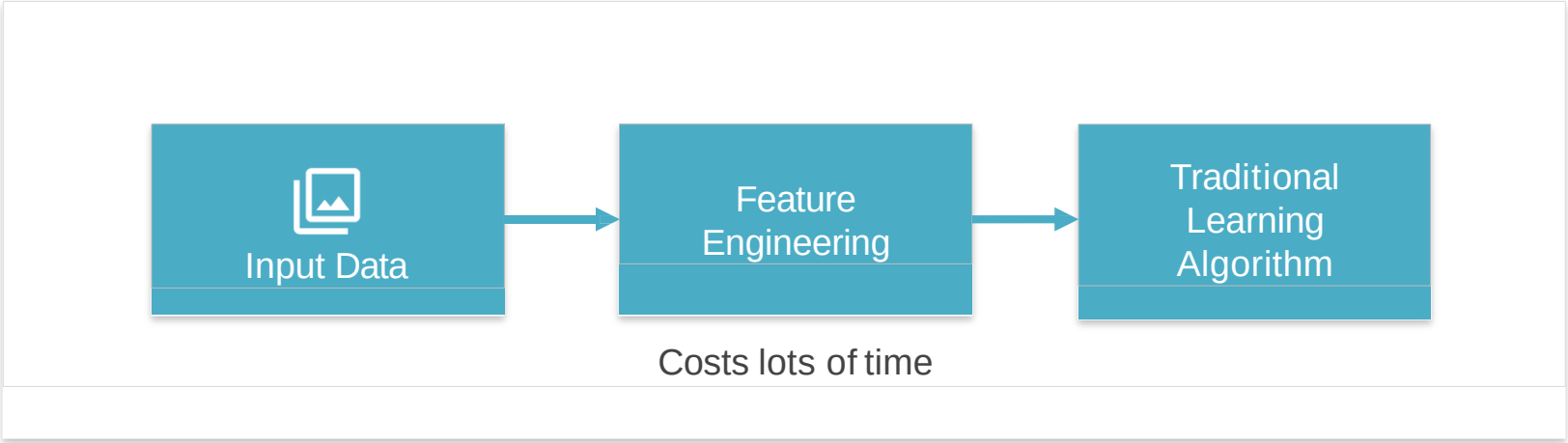
Yann LeCun: New York University & Facebook

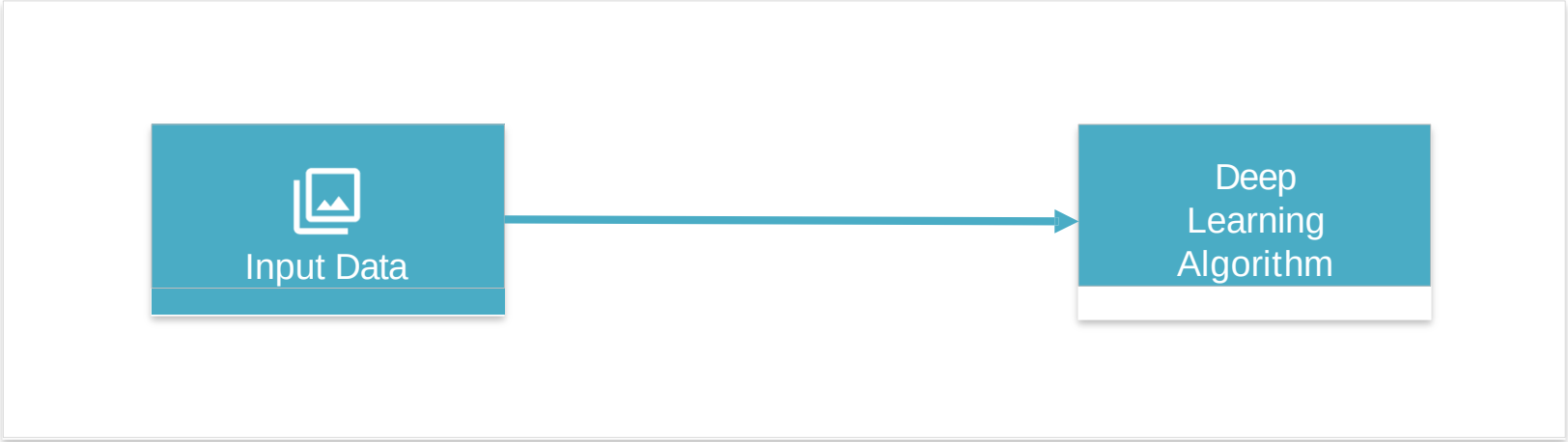
Andrew Ng: Stanford & Baidu

Yoshua Bengio: University of Montreal

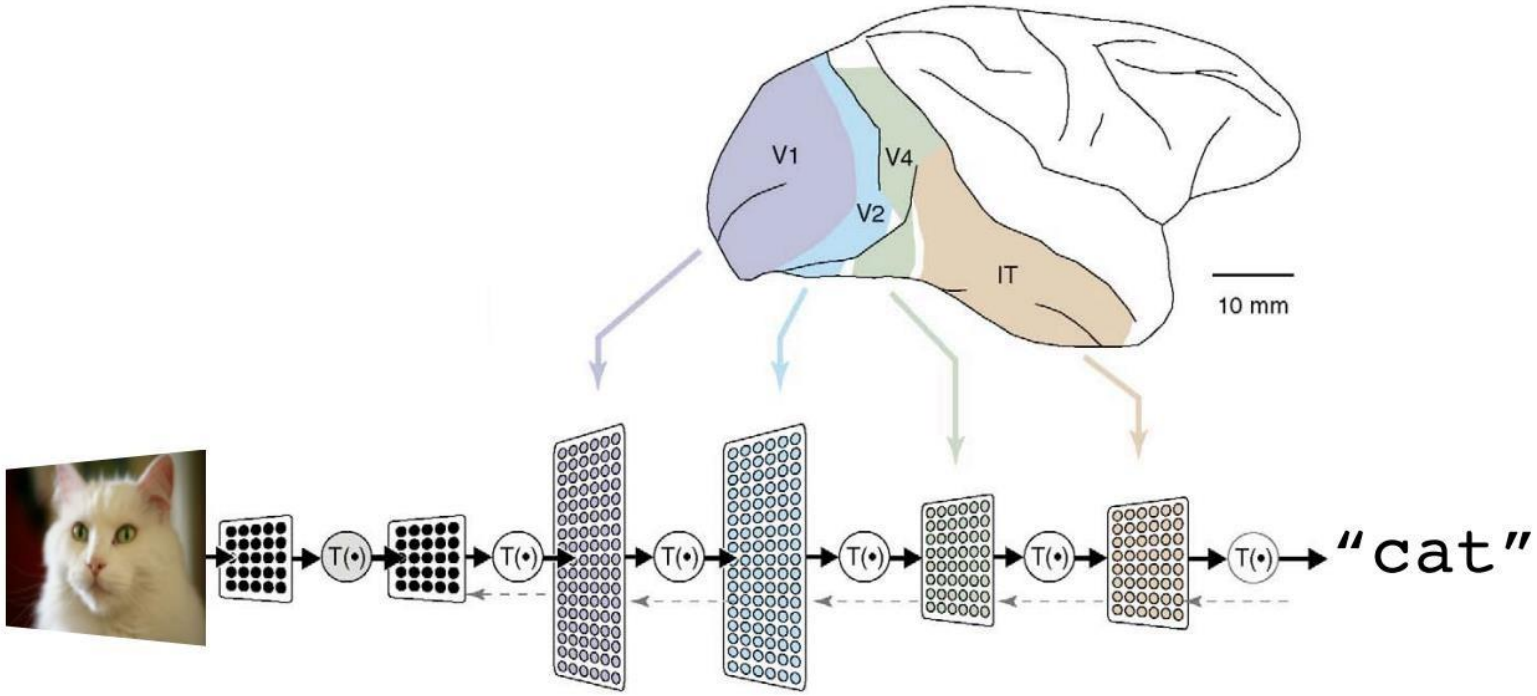
Jürgen Schmidhuber: Swiss AI Lab & NNAISENSE

No more feature engineering



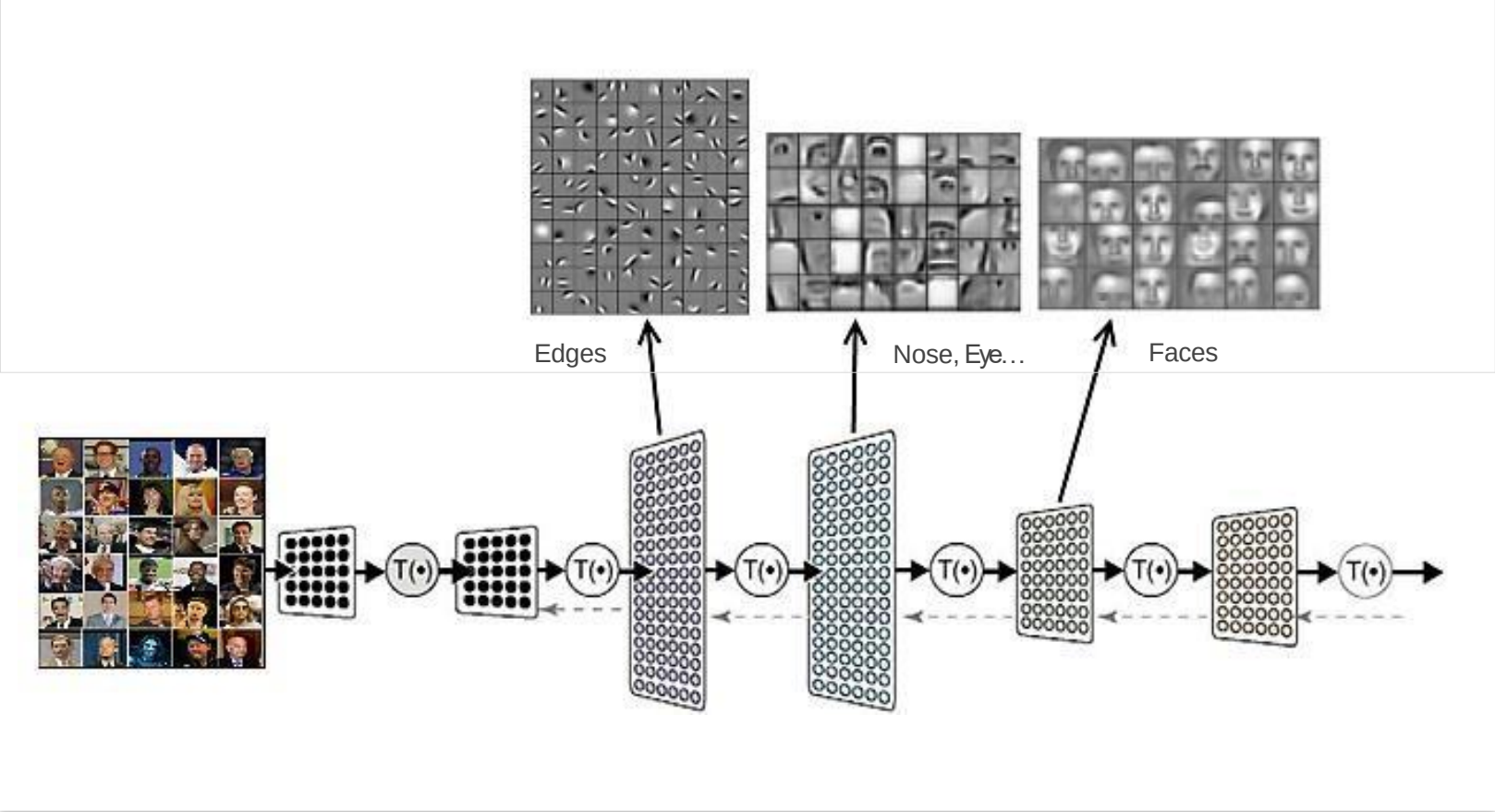


Deep Learning - Architecture

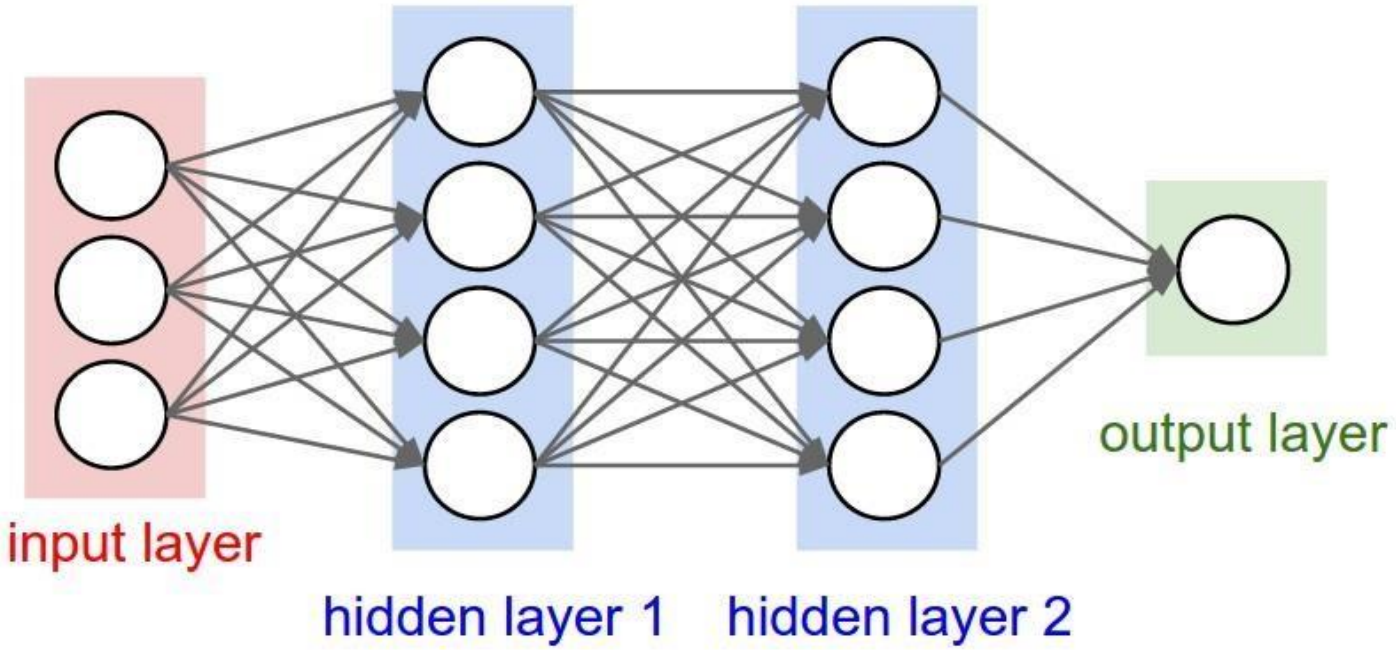


A deep neural network consists of a hierarchy of layers, whereby each layer transforms the input data into more abstract representations (e.g. edge -> nose -> face). The output layer combines those features to make predictions.

Deep Learning - What did it learn?

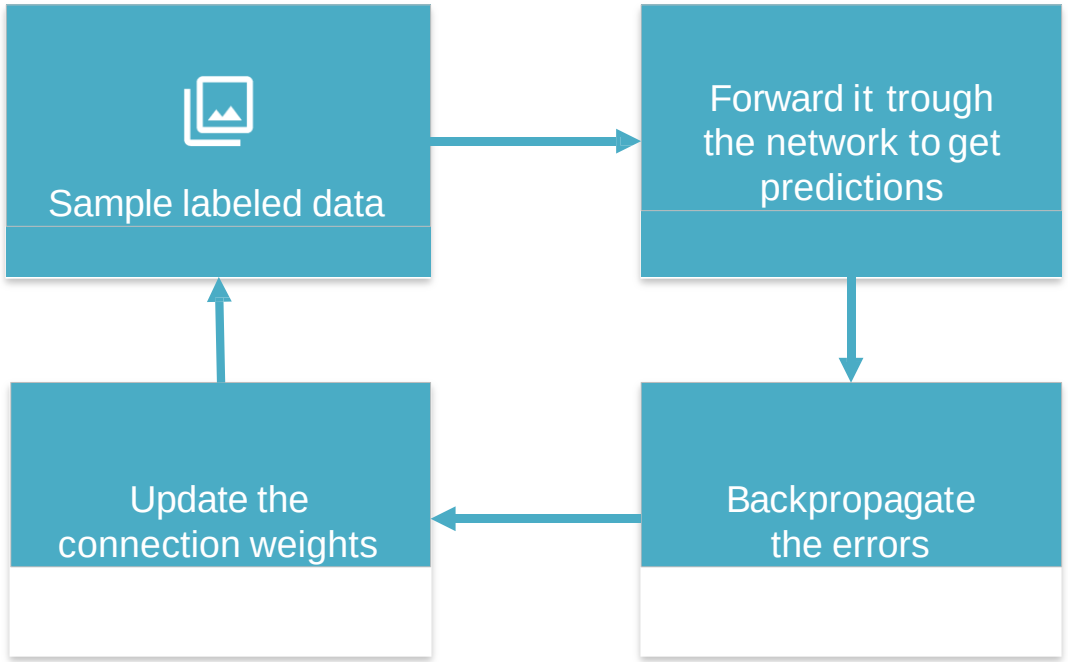


Deep Learning - Artificial Neural Networks



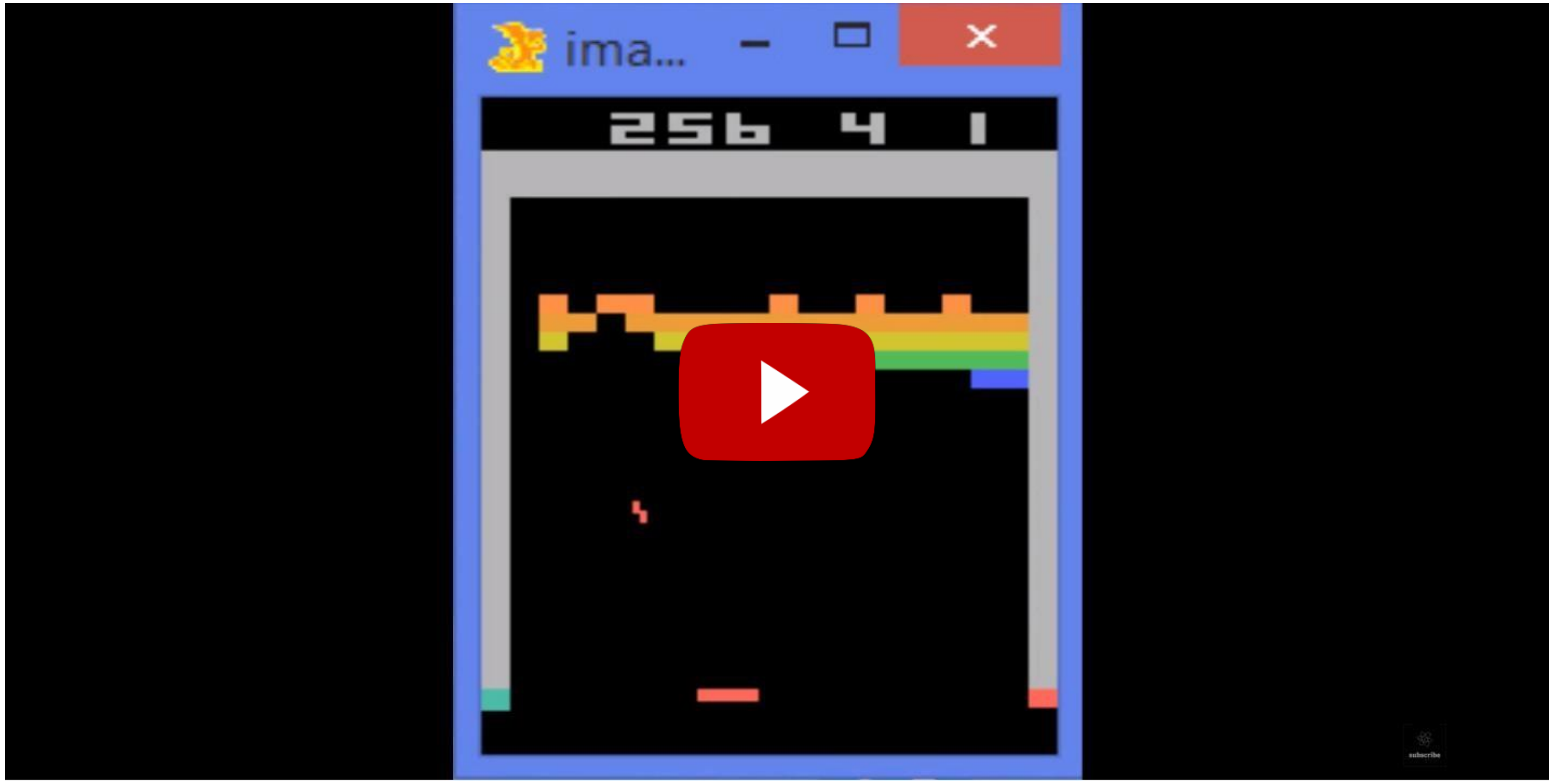
Consists of one input, one output and multiple fully-connected hidden layers in-between. Each layer is represented as a series of neurons and progressively extracts higher and higher-level features of the input until the final layer essentially makes a decision about what the input shows. The more layers the network has, the higher-level features it will learn.

Deep Learning - The Training Process



Learns by generating an error signal that measures the difference between the predictions of the network and the desired values and then using this error signal to change the weights (or parameters) so that predictions get more accurate.

DeepMind Deep Q-Learning



Outperforms humans in over 30 Atari games just by receiving the pixels on the screen with the goal to maximize the score (Reinforcement Learning)

Deep Learning – Usage requirements

Large data set with good quality *(input-output mappings)*

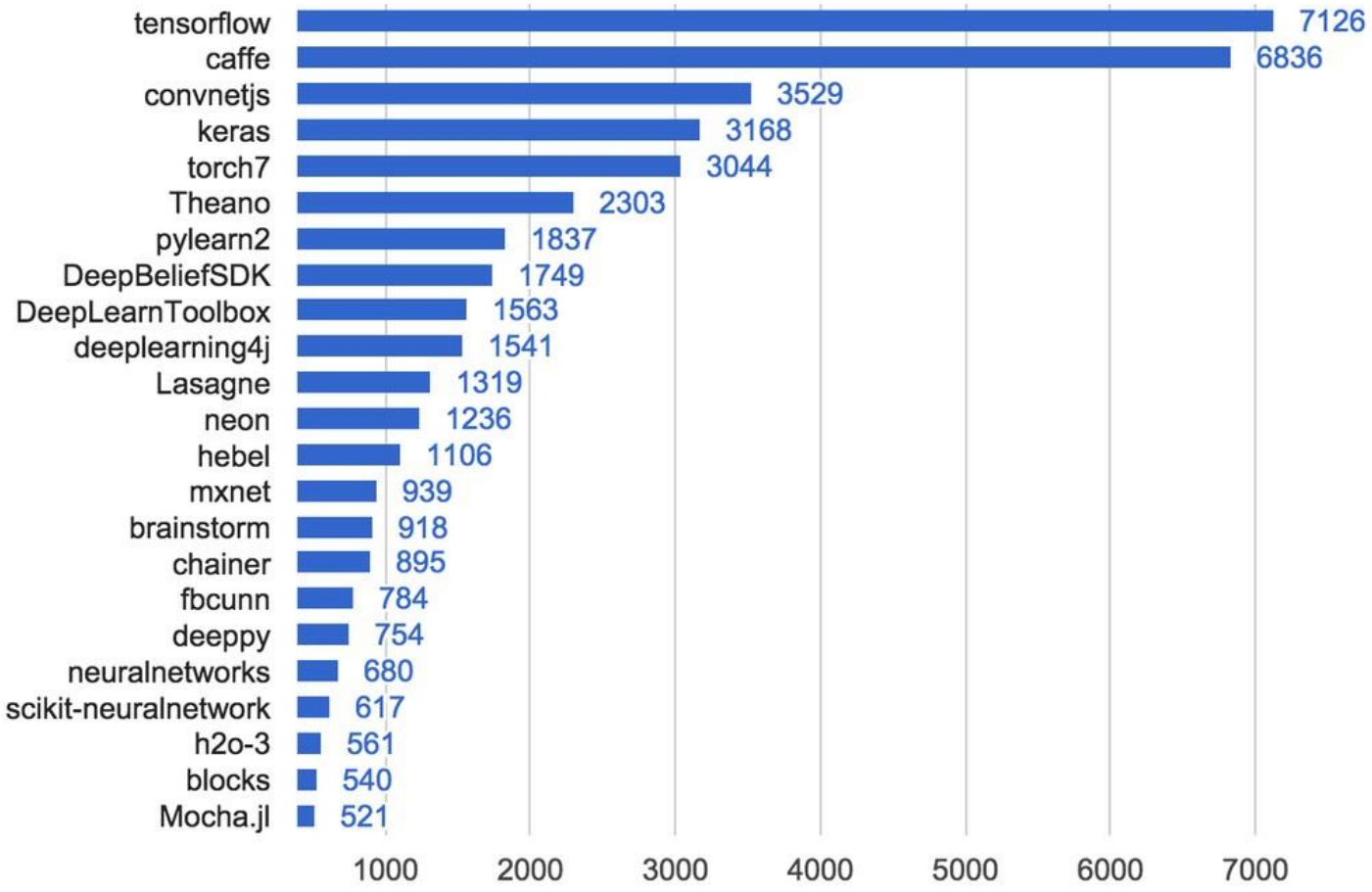
Measurable and describable goals *(define the cost)*

Enough computing power *(AWS GPU Instance)*

Excels in tasks where the basic unit *(pixel, word)* has very little meaning in itself, but the combination of such units has a useful meaning

Deep Learning - Tools

Its all OpenSource



GEOMARKETING

MapInfo ProViewer - [DV BE klanten JORI theoretische pot tov niet klanten JORI theoretische pot]

File Select Options Map Window Help

Legend (Not Responding)

JORI klanten dv

620.000

THEORETISCHE POTENTIALITEIT
WERKELIJKE POTENTIALITEIT
OMZET LAATSTE 12 MAANDEN

Niet JORI klanten dv
THEORETISCHE POTENTIALITEIT

60.000 to 1.000.000 (4)
30.000 to 60.000 (20)
15.000 to 30.000 (0)
0 to 15.000 (0)

JORI delegates gebieden

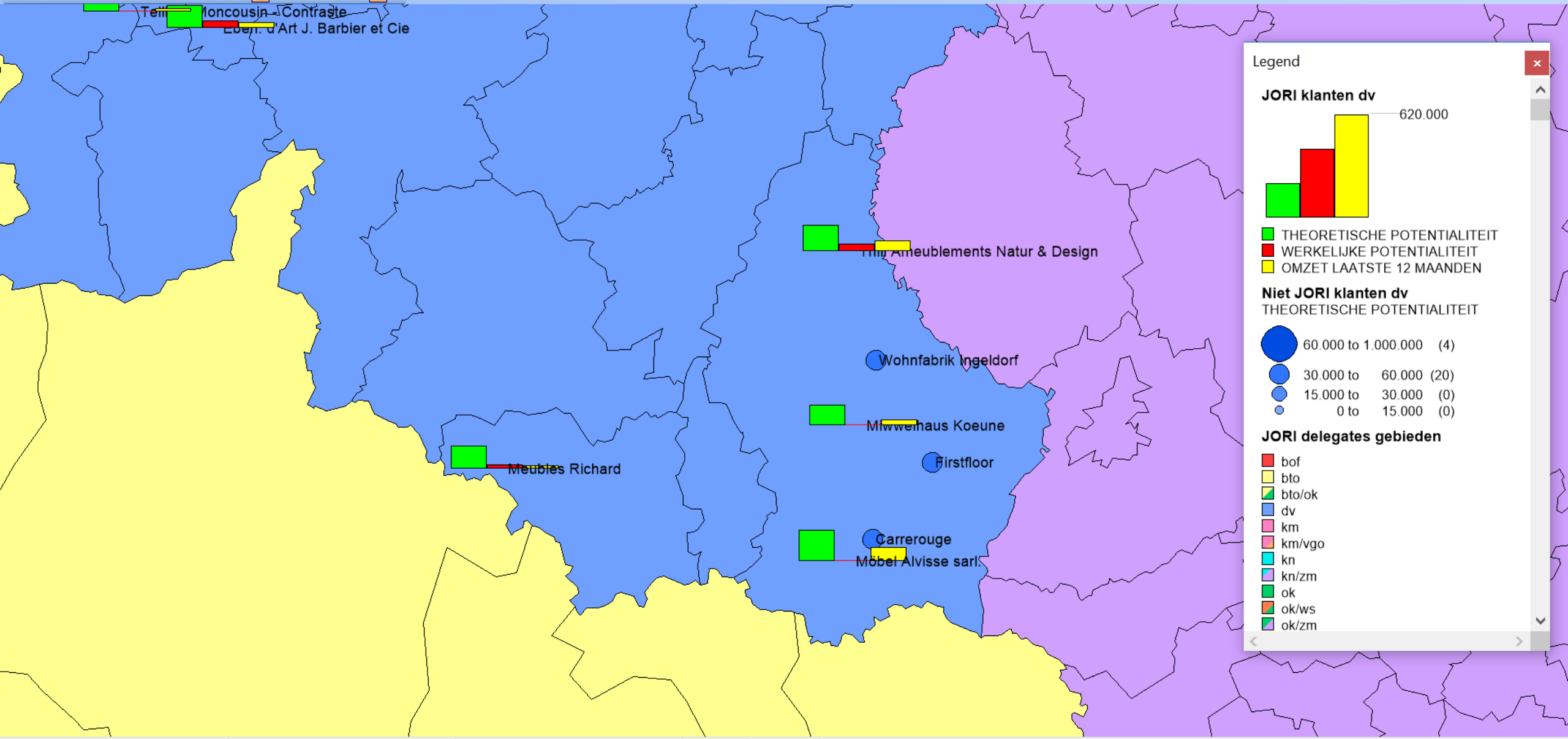
- bof
- bto
- bto/ok
- dv
- km
- km/vgo
- kn
- kn/zm
- ok
- ok/ws
- ok/zm

Zoom: 246,6 km | Editing: None | Selecting: None

NLD BEP 17:55 10/02/2019

MapInfo ProViewer - [DV BE klanten JORI theoretische pot tov niet klanten JORI theoretische pot]

File Select Options Map Window Help



Zoom: 252,1 km | Editing: None | Selecting: eun3_JORI

NLD BEP 17:56 10/02/2019

MapInfo ProViewer - [DV BE klanten JORI theoretische pot tov niet klanten JORI theoretische pot]

File Select Options Map Window Help

Veco Meubel & Interieur
R.R. Devos Interieurs
Master meubel n.v.
Claes Meubelen Dessel
Meubelen Antonis 'n begrip
Verberckmoes Design
Top Interieur
BALO
Colifac
De Bock
Meubelen Van Pee
Interieur Plus
Loft Living
Toro
Vandermeeren Meubelen en Interieur
Stella Meubelen
Sublim Interieur
Peeters - Van Leeuw Meubelen
De Direkteurswoning
Aksent
Living Plus Interior
Zetel- en meubelfabriek De Schoenmaeker
Feelathome
DE RORE Interieur
Pontzeele Interieur
Cockaert Design
Interform
Warson Me
Hermans Interieur/CS Claes
DONUM
't Casteelken Interior Design
Top Interieur
Büro Modern
Meubelen Torrekens
Stolz
Decoration
Boligna Meubelen voor dag en nacht
Leroux Meubelen
Gyselinck
Decovan Home Art
Dominique Rigo
Loncin Interieur
Meubelen Nibem
Makri Meubelen
MAKRI
Vastiau - Godeau
Désiron & Lizen
Inno Design
Meubles Mativa
Vivre Contemporain
Inside Concept
Berhin

Zoom: 246,6 km | Editing: None | Selecting: None

NLD BEP 18:28 10/02/2019

MapInfo ProViewer - [OK DE CH klanten JORI theoretische pot tov niet klanten JORI theoretische pot]

File Select Options Map Window Help

Wohnland Breitwieser
Axel Walther Wohnbedarf GmbH
Möbel Ehrmann
Schmezer Einrichtungshaus
Möbel Ehrmann
Art Tisch GmbH
Weber Einrichtungen
Möbel Glib
Möbel-Center Bierstorfer
Einrichtungshaus Pomm
Jäger Einrichtungen
Stork - Die Einrichtung
Paul Feederle GmbH
Blow Out GmbH
Atrium Möbel und Design
Hofmeister
Schweizer - Die Einrichter
Wohnfabrik pappe La Pappe
Die Einrichtung Kleemann
Weimer das
Benz Einrichtungen GmbH
Kübel + Partner - Die neue Einrichtung
Möbel Koch Einrichtungshaus
renz möbel
Architare - Barbara Benz Einrichten GmbH & C
Kruss Einrichtungen
Lukaszewitz Einrichten & Wohnen
Möbel Bühler Einrichtungshaus
Dobergo Büromöbelfabrik GmbH

Legen
JOR
T
W
O
Niet
THE
JOR
b
b
b
d
k
k
k
k
o
o

Zoom: 257,2 km | Editing: None | Selecting: None

NLD BEP 18:31 10/02/2019

NEXT STEPS

Next steps – general info

- https://nodered.org/
- https://www.youtube.com/watch?v=O9sCM_l7SiI

Next steps – courses and IBM i info

- https://www.ibm.com/developerworks/ibmi/library/i-running-node-red/index.html
- https://www.ibm.com/developerworks/cloud/library/cl-rtchat-app/index.html
- https://www.ibm.com/developerworks/ibmi/library/i-it-helpdesk-chatbot/
- https://developer.ibm.com/courses/all/node-red-basics-bots/?course=begin#4069
- https://www.ibm.com/developerworks/ibmi/library/i-ile-rpg-cloud-integration/

THANK YOU FOR YOUR ATTENTION