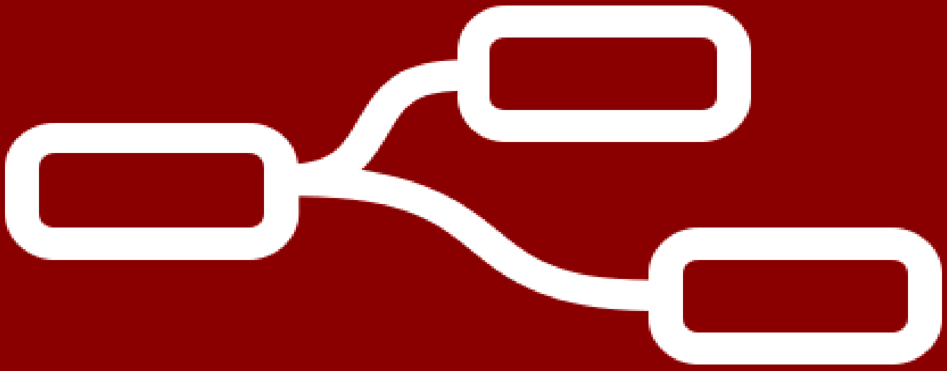
MAKERDEMY

THE ONE WITH THE N.O.D.E-R.E.D. PRIMER



R.A.C.H.E.N. R.A.V.I.C.H.A.N.D.R.A.N.

MAKERDEMY

THE ONE WITH THE N.O.D.E-R.E.D. PRIMER

GETTING NODE-READY

R.A.C.H.E.N. R.A.V.I.C.H.A.N.D.R.A.N.

THE ONE WITH ALL THE C.O.N.T.E.N.T.S.

1. THE ONE WITH THE AVENGERS

Cloud9 supermarket boasts a range of daily products. If you had been to this store, you'd be amazed by its technology. The shop automatically opens at 6 AM and the computers in their Headquarters control the Air Conditioning temperature so that customers would feel comfy in the store. Recently, when Avengers Infinity war teaser was released and was trending on Twitter, the company immediately rolled their Infinity War Posters into the store in the next minute. Cool! Cloud9 really seems to make their customers feel on cloud nine. But, how? How do they do it?

IoT. Internet of Things. Simply put, you get information from any source you can, figure out how to use that data and put that data into use, all of which is done with the internet as a transporting medium for data.

Let's figure out how Cloud9 store did it. But, you are completely new to IoT and too naïve to code thousands of lines. So, how are you going to do it?

Node-RED comes to save our day...

2. THE ONE WITH THE INSTALLATION

Let's install Node-RED. Follow the steps given below:

Step 1: Installing Node.js

Go to the link: Node.js Installation and download Node.js according to your computer specifications. Then, install Node.js (self-explanatory.)

Step 2: Installing Node-RED for Windows

Open Node.js command prompt, which you've installed just now, and type:

npm install –g --unsafe-perm node-red

The installation should be complete in few minutes. For further information, visit: Installation Instructions

While Node-RED is installing, read this!

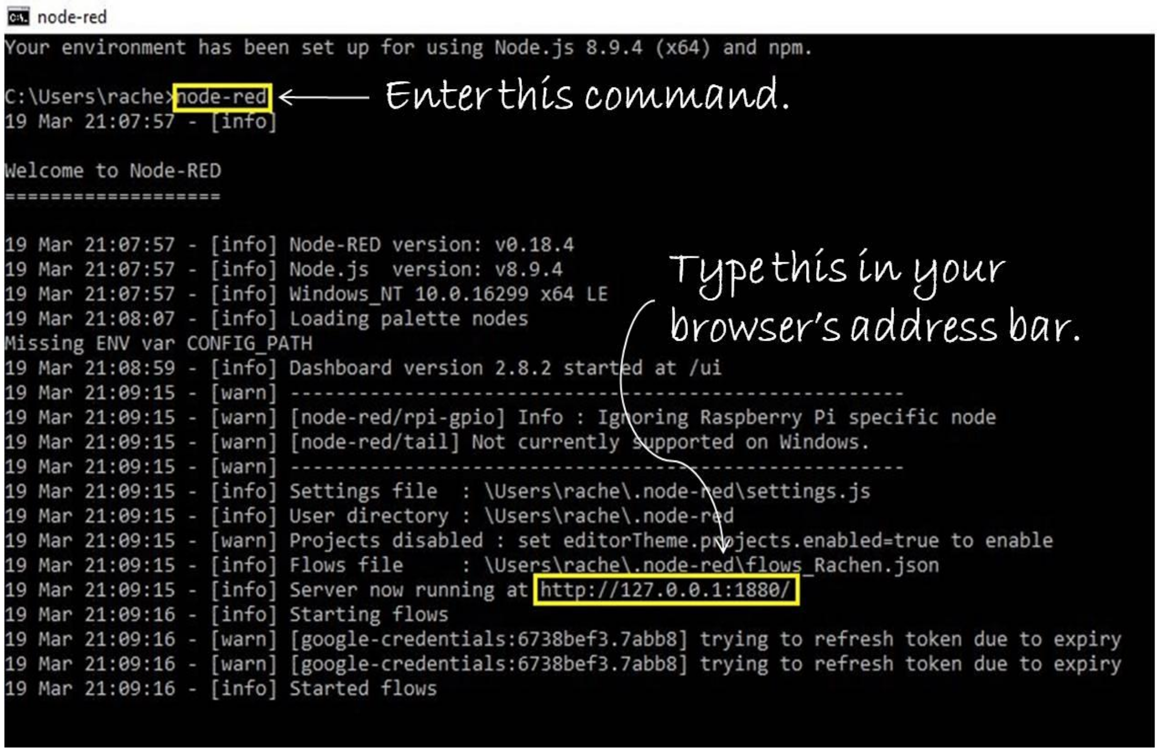
Node-RED is an open source project which was created by some intelligent folks at IBM. It is a visual (flow based) programming tool where you drag and drop the programming elements you need, instead of typing hundreds of lines of codes and remembering the syntaxes.

It can be integrated with your IoT projects and use many of its inbuilt IoT functions without worrying about re-inventing the wheel. Just figure out what you need for your application, drag and drop the elements required, do some changes and there you go!

Step 3: Launching Node-RED

Now, in the Node.js command prompt, enter:

node-red



```
node-red
Your environment has been set up for using Node.js 8.9.4 (x64) and npm.

C:\Users\rache>node-red
19 Mar 21:07:57 - [info]

Welcome to Node-RED
===================

19 Mar 21:07:57 - [info] Node-RED version: v0.18.4
19 Mar 21:07:57 - [info] Node.js  version: v8.9.4
19 Mar 21:07:57 - [info] Windows_NT 10.0.16299 x64 LE
19 Mar 21:08:07 - [info] Loading palette nodes
Missing ENV var CONFIG_PATH
19 Mar 21:08:59 - [info] Dashboard version 2.8.2 started at /ui
19 Mar 21:09:15 - [warn] ------------------------------------------------------
19 Mar 21:09:15 - [warn] [node-red/rpi-gpio] Info : Ignoring Raspberry Pi specific node
19 Mar 21:09:15 - [warn] [node-red/tail] Not currently supported on Windows.
19 Mar 21:09:15 - [warn] ------------------------------------------------------
19 Mar 21:09:15 - [info] Settings file  : \Users\rache\.node-red\settings.js
19 Mar 21:09:15 - [info] User directory : \Users\rache\.node-red
19 Mar 21:09:15 - [warn] Projects disabled : set editorTheme.projects.enabled=true to enable
19 Mar 21:09:15 - [info] Flows file     : \Users\rache\.node-red\flows_Rachen.json
19 Mar 21:09:15 - [info] Server now running at http://127.0.0.1:1880/
19 Mar 21:09:16 - [info] Starting flows
19 Mar 21:09:16 - [warn] [google-credentials:6738bef3.7abb8] trying to refresh token due to expiry
19 Mar 21:09:16 - [warn] [google-credentials:6738bef3.7abb8] trying to refresh token due to expiry
19 Mar 21:09:16 - [info] Started flows
```

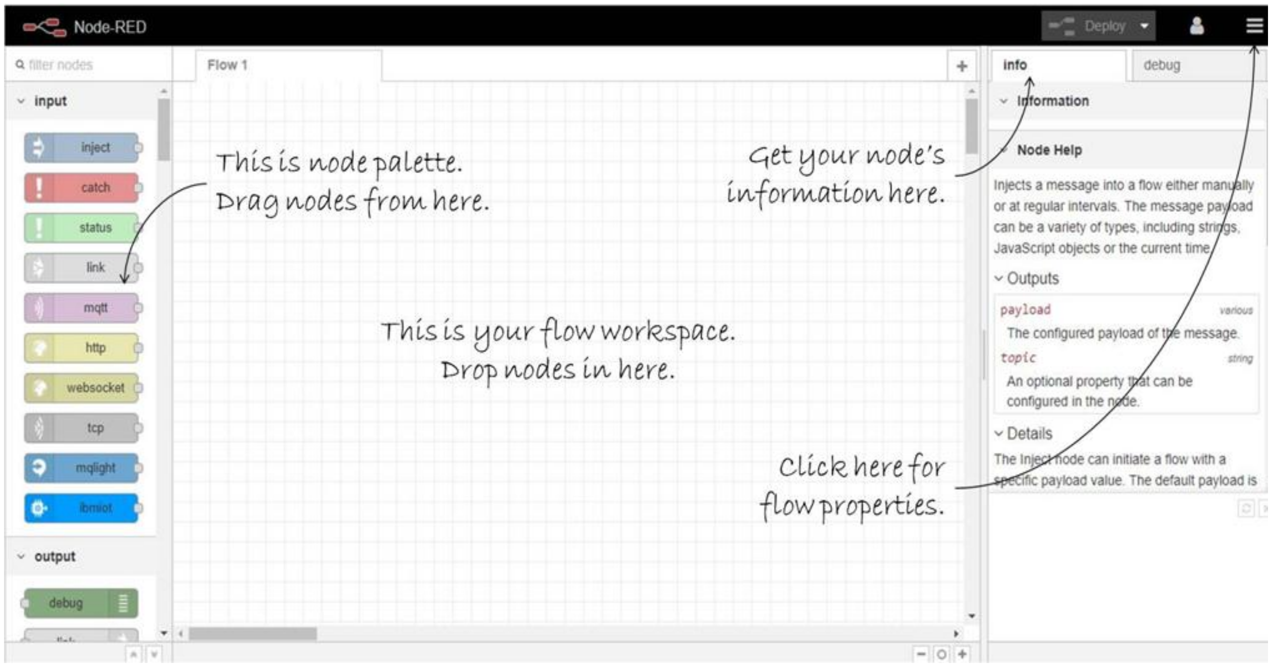
Fig. 2.1 Launching your Node-RED

It might take few minutes to load the Node-RED environment, but you'll know that all is up when it shows 'Started flows' in the cmd. Now, open your web browser and type the URL, given in the command prompt, in your address bar (marked in the yellow box in Fig. 2.1.)

Now, your Node-RED is ready to be used.

3. THE ONE WITH THE FLOW

Once Node-RED is open, you'll find a window as shown in Fig. 3.1.



Injects a message into a flow either manually or at regular intervals. The message payload can be a variety of types, including strings, JavaScript objects or the current time.

Outputs

payload various
The configured payload of the message.
topic string
An optional property that can be configured in the node.

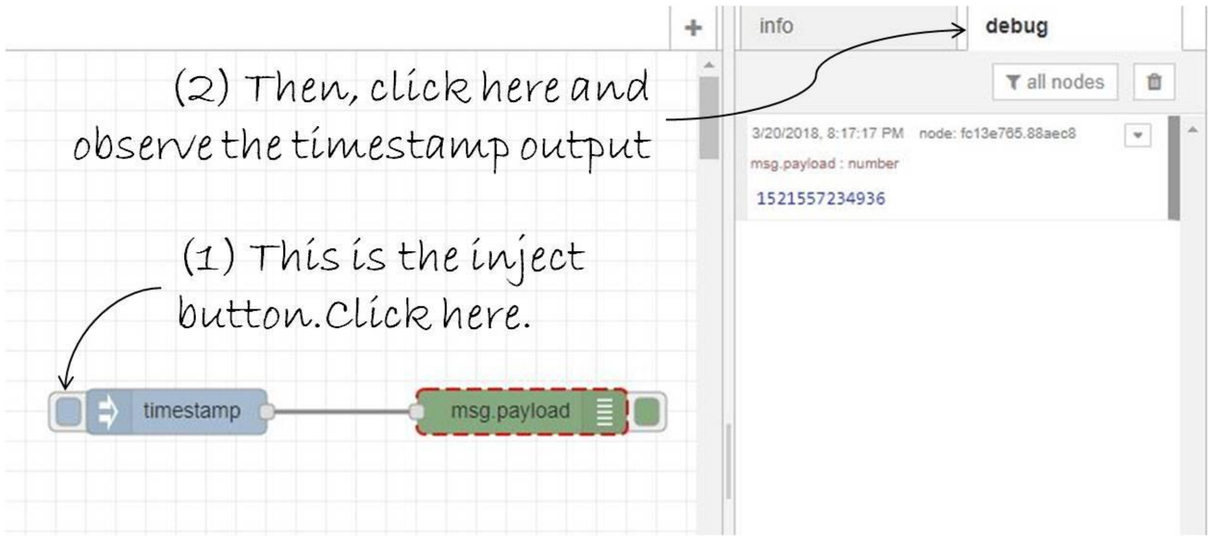
Details

The Inject node can initiate a flow with a specific payload value. The default payload is

Fig. 3.1. Workspace of Node-RED

The coloured boxes are called nodes. There are grey dots in each node. To connect two nodes, start from the grey dot on any one of the nodes to the grey dot on the other. As an example, from the node palette, drag and drop inject and debug node into the flow workspace. Connect the two nodes as in Fig. 3.2. Click the 'Deploy' button at the top right corner.

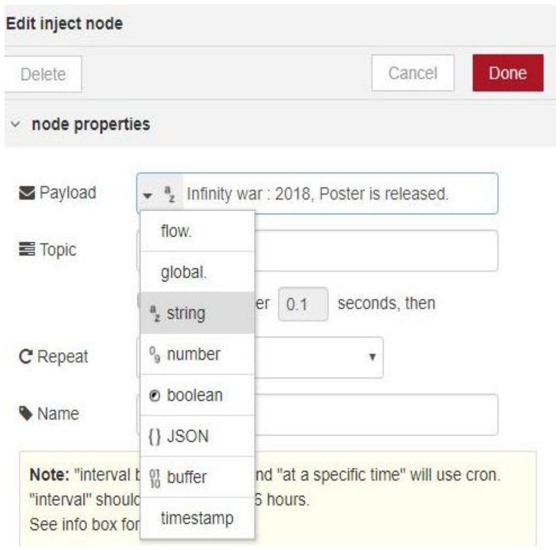
Press the square button to the left of the inject node, called the inject button. Now, open the debug tab in the right pane. Ha! You can see that the debug tab carries the date and time of pressing the button (called injecting) as in Fig. 3.2.


(2) Then, click here and observe the timestamp output
(1) This is the inject button. Click here.
info
debug
all nodes
timestamp
msg.payload

3/20/2018, 8:17:17 PM node: fc13e765.88aec8
msg.payload : number
1521557234936

Fig. 3.2. Using **inject** *and* **debug** *nodes*

The output, actually, shows a strangely structured quirky number called the timestamp. It is the time elapsed in milliseconds from January 1st, 1970 to the instant you've injected the node. Now, click the inject node twice, change the Payload property to a string, enter 'Infinity war : 2018, Poster is released.' and click 'Done' (Fig. 3.3.) Deploy your flow. Press the inject button and check the output in debug tab. You'll receive the input sentence as output in your debug tab.



Payload: Infinity war : 2018, Poster is released.

…er 0.1 seconds, then

Note: "interval b… …nd "at a specific time" will use cron.
"interval" should… …6 hours.
See info box for…

Fig. 3.3. Configuring **inject** *node properties*

In Node-RED all the data are transmitted in a variable 'msg' which is an 'object' variable. To understand objects, consider the menu below:

Ice-cream	
Ice-cream type - Key or Property	**Price (in Cloud9 store credits) - Value**
Vanilla	20
Butter Scotch	30
Chocolate	25
Sundae	40

If ice-cream is to be used as an object, then the table would look like this:

```
ice_cream = {
               vanilla : 20,
               butter_scotch : 30,
               chocolate : 25,
               sundae : 40
             }
```

If you want to find out the price of sundae type of ice-cream, then use:

```
price = ice_cream.sundae
```

Similarly, msg.payload is the variable which generally holds your data, through which most of the node puts or gets the information.

4. THE ONE WHERE YOU PUT ALL TOGETHER

Using function node, you can cook your own codes. Function node is similar to functions in other programming languages. When you are going to use the same code again and again, you use function node. You can process your data in it. They are written in JavaScript (JS).

Drag a function node and place it in between the inject and debug node. You will see the connecting wire turning into dashed wire when you drag it in between the other two nodes and will automatically rewire when you release the node as in Fig 4.1.

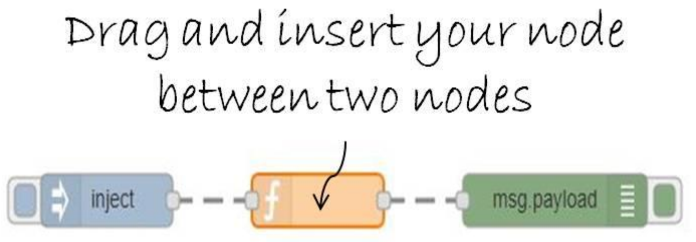
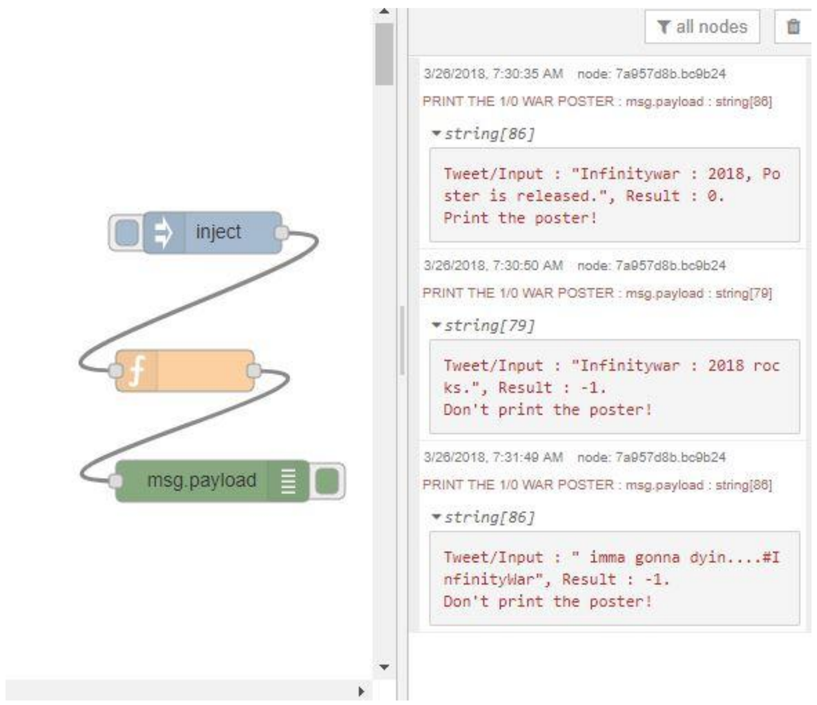


Fig. 4.1. When inserting a node between two other nodes, the connecting wires become dashed

Double click the function node, paste the code from the file 'Avengers_Poster.js' into the node and click 'Done'. You can vary the string in the inject node one by one as in Table 1 and observe the output in the debug tab by clicking the inject button after deploying. The outputs for various inputs are given in Fig. 4.2.

Table 1 : Inputs

1. Infinitywar : 2018, Poster is released.
2. Imma gonna dyin....#InfinityWar
3. #InfinityWar posters released



```
3/26/2018, 7:30:35 AM   node: 7a957d8b.bc9b24
PRINT THE 1/0 WAR POSTER : msg.payload : string[86]
▾string[86]
Tweet/Input : "Infinitywar : 2018, Po
ster is released.", Result : 0.
Print the poster!

3/26/2018, 7:30:50 AM   node: 7a957d8b.bc9b24
PRINT THE 1/0 WAR POSTER : msg.payload : string[79]
▾string[79]
Tweet/Input : "Infinitywar : 2018 roc
ks.", Result : -1.
Don't print the poster!

3/26/2018, 7:31:49 AM   node: 7a957d8b.bc9b24
PRINT THE 1/0 WAR POSTER : msg.payload : string[86]
▾string[86]
Tweet/Input : " imma gonna dyin....#I
nfinityWar", Result : -1.
Don't print the poster!
```

Fig. 4.2. Using Avengers_Poster.js in your flow

Let's find out how we have defined our `function` node.

```
var exp = /Infinity.*war.*poster.*released/i;
msg.search_result = msg.payload.search(exp);
```

The first line is some odd looking data which is called as a regular expression. They represent the pattern which is to be matched. To understand about its working, go to this link. In the flags tabs, uncheck the global option and select the case-insensitive option as in Fig. 4.3.

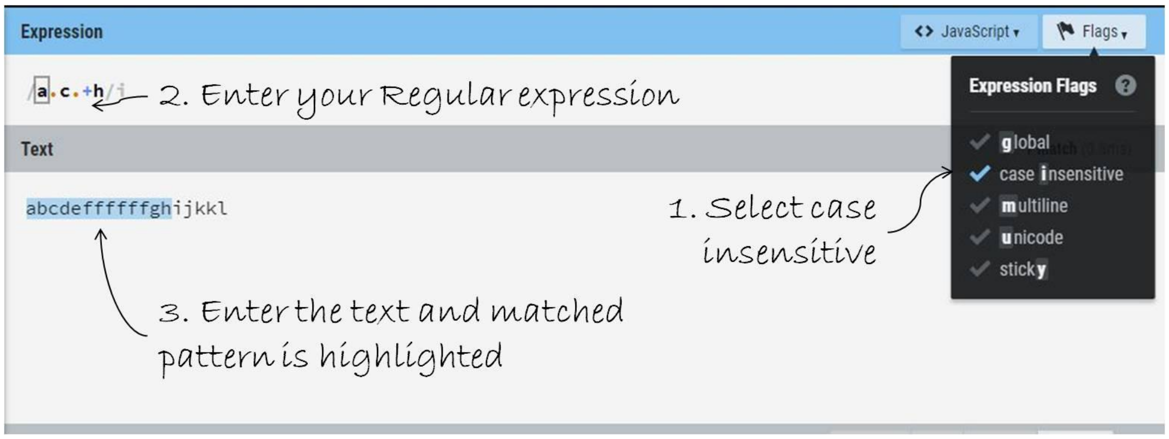


Fig. 4.3. Figuring out what regular expressions are.

Type the following in the expression:

```
/a.c.+h/i
```

Type any text you wish. For example, “abcdefffffghijkkl”. You can see that only part of the text is selected as ‘abcdefffffghijkkl’.

This is because, ‘.’ in the expression indicates any characters except line breaks. The letters represent the exact characters to match. ‘+’ indicates that multiple occurrences of its preceding character are allowed, i.e. ‘+’ matches 1 or more occurrence of the character preceding it. Since it is preceded by ‘.’ as in ‘.+’, multiple occurrences of any character are possible i.e, for ‘c.+’, chhhhh, cah, ch are all valid patterns. The ‘i’ at the end indicates that the pattern does not care for upper or lower cases. Try using ‘*’ instead of ‘+’ where it matches 0 or more occurrence of the character preceding it.

Thus, any pattern can be represented by these regular expressions. The pattern required is enclosed within ‘/ /’ followed by flags (here ‘i’, case insensitive, is used.) Try typing the expression as given in variable exp and type in any sentence you want in the link. Some of the valid strings matching the pattern are:

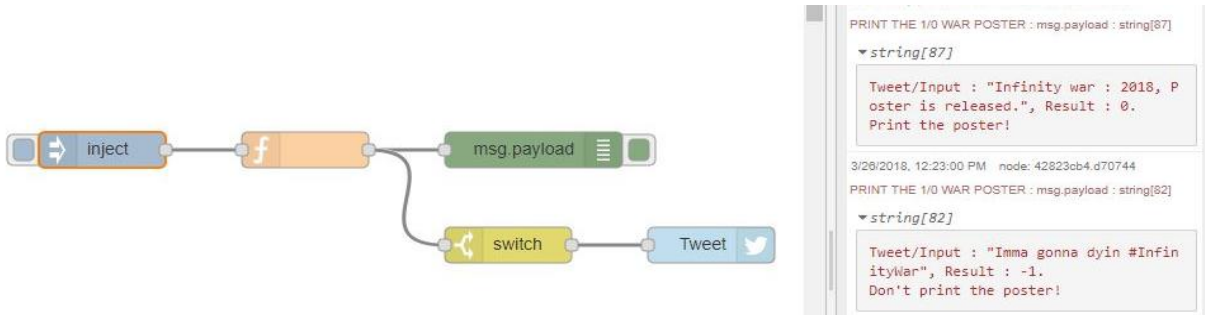
1. Infinity war!!!! Posters are released.
2. Ha Ha! Infinityyyyyyy Warrrrrr Posters are released!
3. infinity WAR POSters RELEasED
4. #InfinityWar posters released.

Thus, if we have any of the above words matching with the tweet, we can say that the posters are released.

Note: This pattern search will also match 'InfinityWar posters will be released', which in reality means the posters are yet to be released. For reducing complexity, I haven't included this condition. Try including that expression too.

The second line of the code does the work of searching for the pattern in the payload message and returns the index position at which the pattern matches. If no pattern match is found, it returns -1. This result is stored in `search_result` value. `search_result` is checked using 'if' condition to determine whether to print the poster or not. The results are then stored in `msg.payload` and returned as `msg` object.

Now let us add a switch node to the function node and then connect a tweet out node to switch node as in Fig. 4.4.



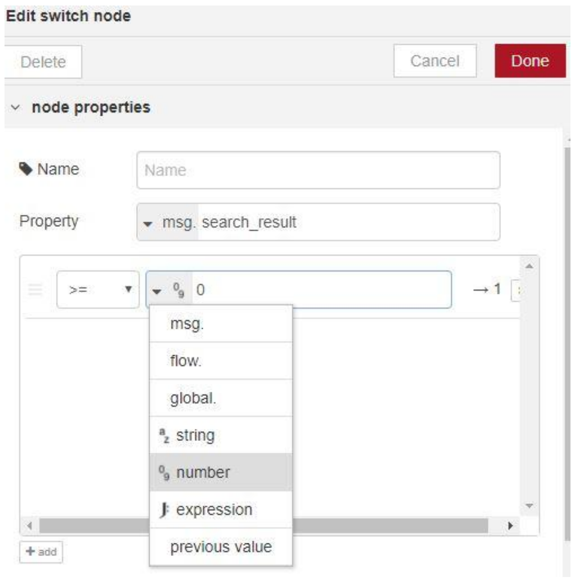
```
PRINT THE 1/0 WAR POSTER : msg.payload : string[87]
▾string[87]
Tweet/Input : "Infinity war : 2018, P
oster is released.", Result : 0.
Print the poster!

3/26/2018, 12:23:00 PM   node: 42823cb4.d70744
PRINT THE 1/0 WAR POSTER : msg.payload : string[82]
▾string[82]
Tweet/Input : "Imma gonna dyin #Infin
ityWar", Result : -1.
Don't print the poster!
```

*Fig. 4.4. Adding **switch** and **tweet out** node to your flow*

Now double click the switch node, enter the properties as in Fig. 4.5 (next page) and click 'Done'.

This node allows `msg.payload` to be passed to the twitter node only if `msg.search_result` is greater than or equal to 0.

Double click the twitter node. Click the pencil icon to add your Twitter credentials (Your twitter account's Username and Password) and sign in to your twitter account as in Fig. 4.6 (next page.)


Edit switch node

Delete | Cancel | Done

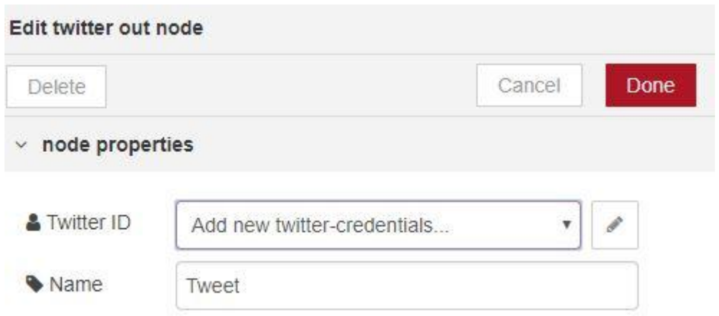
node properties

Name: Name

Property: msg. search_result

>= | 0 → 1

Fig. 4.5. Configuring **switch** *node properties*


Edit twitter out node

Delete | Cancel | Done

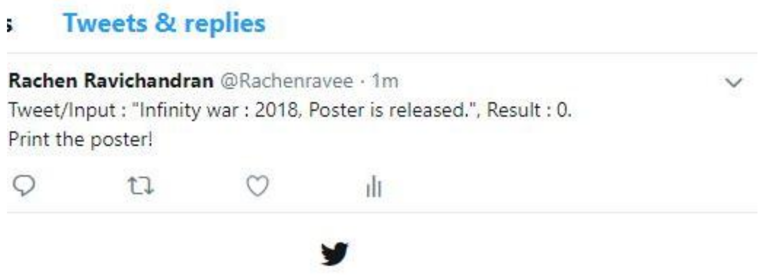
node properties

Twitter ID: Add new twitter-credentials...

Name: Tweet

Fig. 4.6. Entering your twitter credentials in **twitter out** *node*

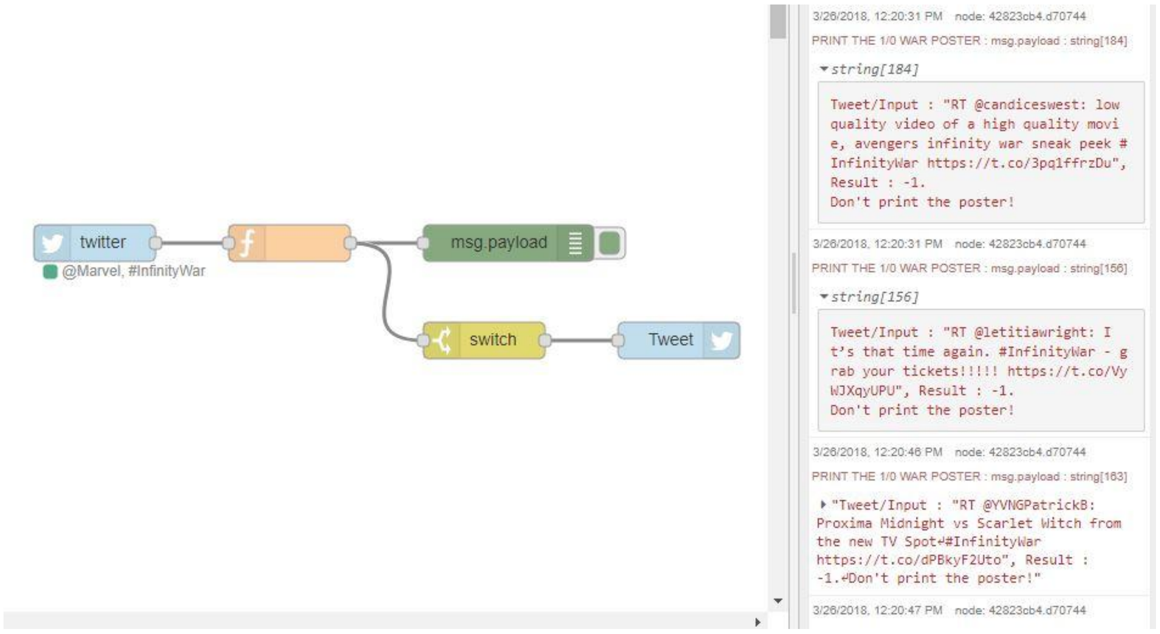
Now click the inject button after deploying and you will get a tweet in your account if the poster is to be printed as in Fig. 4.7. Try for the inputs in Table 1 and observe the output in your twitter account and debug tab.


Tweets & replies

Rachen Ravichandran @Rachenravee · 1m
Tweet/Input : "Infinity war : 2018, Poster is released.", Result : 0.
Print the poster!

Fig. 4.7. A tweet is posted in your twitter account when the poster is to be printed

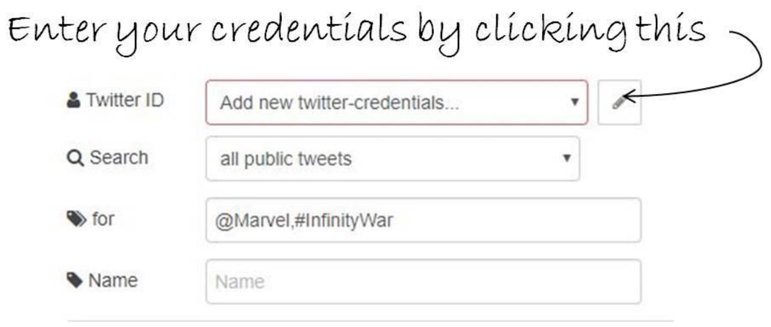
As a final step, replace the inject node with the twitter in node (Fig. 4.8.) Double click the twitter node. Click the pencil icon to add your twitter account credentials by signing in.



```
3/26/2018, 12:20:31 PM   node: 42823cb4.d70744
PRINT THE 1/0 WAR POSTER : msg.payload : string[184]
▾string[184]
Tweet/Input : "RT @candiceswest: low
quality video of a high quality movi
e, avengers infinity war sneak peek #
InfinityWar https://t.co/3pq1ffrzDu",
Result : -1.
Don't print the poster!
3/26/2018, 12:20:31 PM   node: 42823cb4.d70744
PRINT THE 1/0 WAR POSTER : msg.payload : string[156]
▾string[156]
Tweet/Input : "RT @letitiawright: I
t's that time again. #InfinityWar - g
rab your tickets!!!!! https://t.co/Vy
WJXqyUPU", Result : -1.
Don't print the poster!
3/26/2018, 12:20:46 PM   node: 42823cb4.d70744
PRINT THE 1/0 WAR POSTER : msg.payload : string[163]
▸"Tweet/Input : "RT @YVNGPatrickB:
Proxima Midnight vs Scarlet Witch from
the new TV Spot↵#InfinityWar
https://t.co/dPBkyF2Uto", Result :
-1.↵Don't print the poster!"
3/26/2018, 12:20:47 PM   node: 42823cb4.d70744
```

*Fig. 4.8. Replacing **inject** node (from Fig. 4.4) with **twitter in** node*

Then, add the tags and users you want to receive tweets from, separated by ',' as in Fig. 4.9 (here '@Marvel,#InfinityWar'.) Deploy the flow and you'll receive the output as shown in Fig. 4.8.



Twitter ID	Add new twitter-credentials...
Search	all public tweets
for	@Marvel,#InfinityWar
Name	Name

Fig. 4.9. Entering your twitter credentials

If you were to write the code for the twitter node yourself - to connect with it, request for data, validate your account, exchange data, and so on - it would have taken days together to write one. With Node-RED, all the complexities are bottled away within a teeny box for easy implementation and immediate deployment of your IoT projects.

Congratulations on cracking Cloud9 secret!