

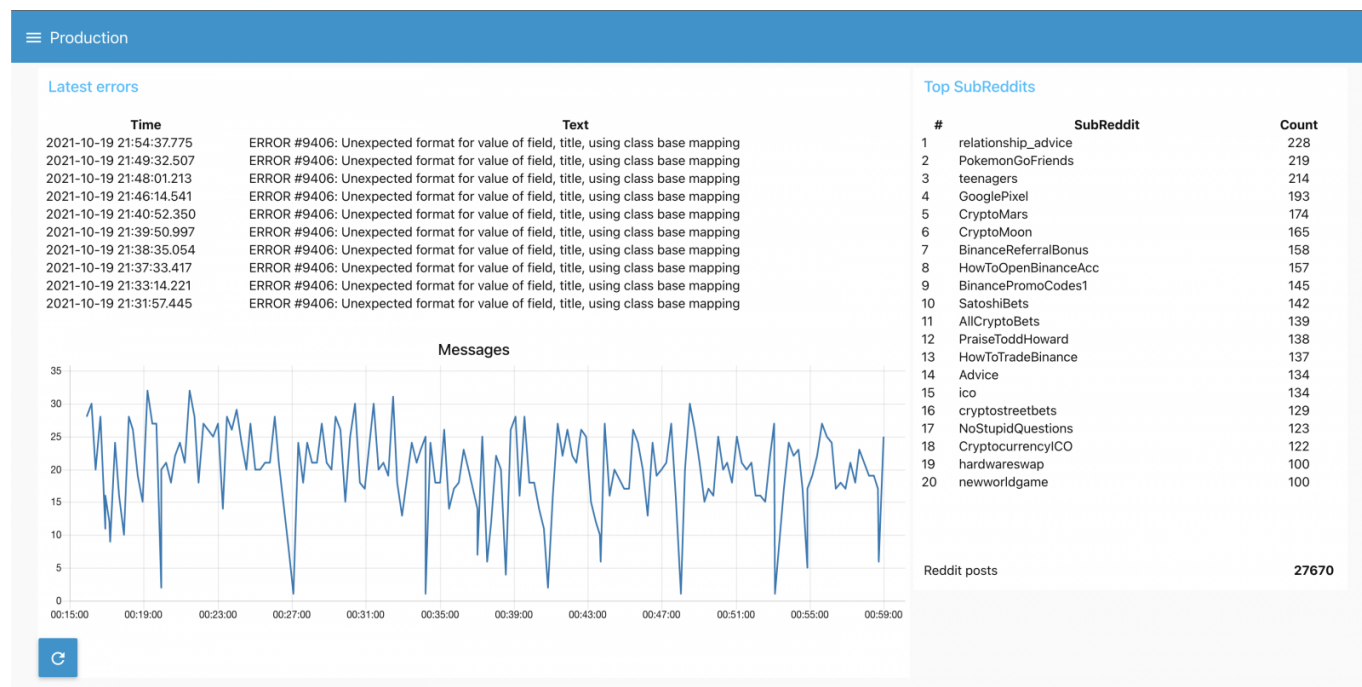
## Article

[Dmitry Maslennikov](#) · Oct 19, 2021 3m read

[Open Exchange](#)

# Monitor production with Node-RED

I would like to demonstrate how you can create monitoring of Production with [Node-RED](#), and get a dashboard like this.

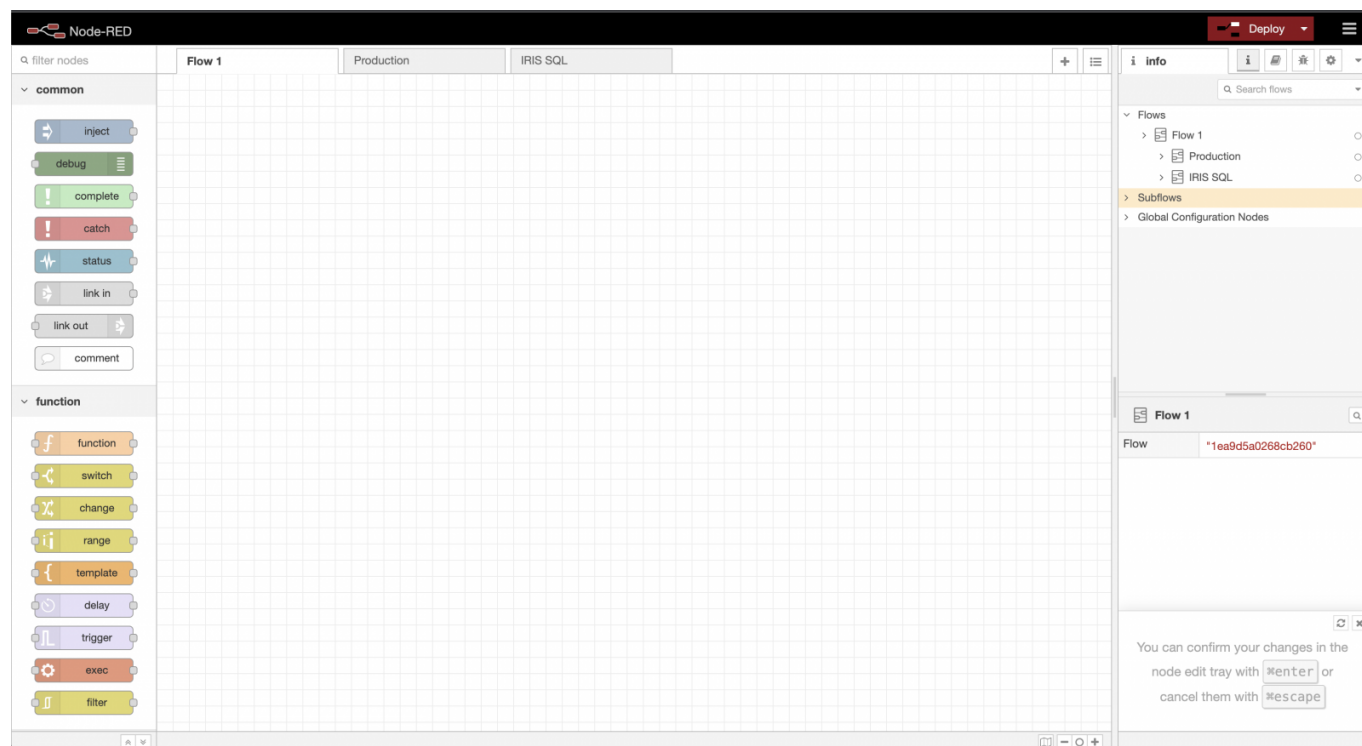


First of all, you would need to run Node-RED. It's possible to install it locally or, run it with Docker. Official Docker images for Node-RED based on Alpine, and may not work correctly with IRIS, yet. So, please use image `caretdev/node-red`

```
docker run -it -p 1880:1880 -v myNodeREDdata:/data --name mynodered caretdev/node-red
```

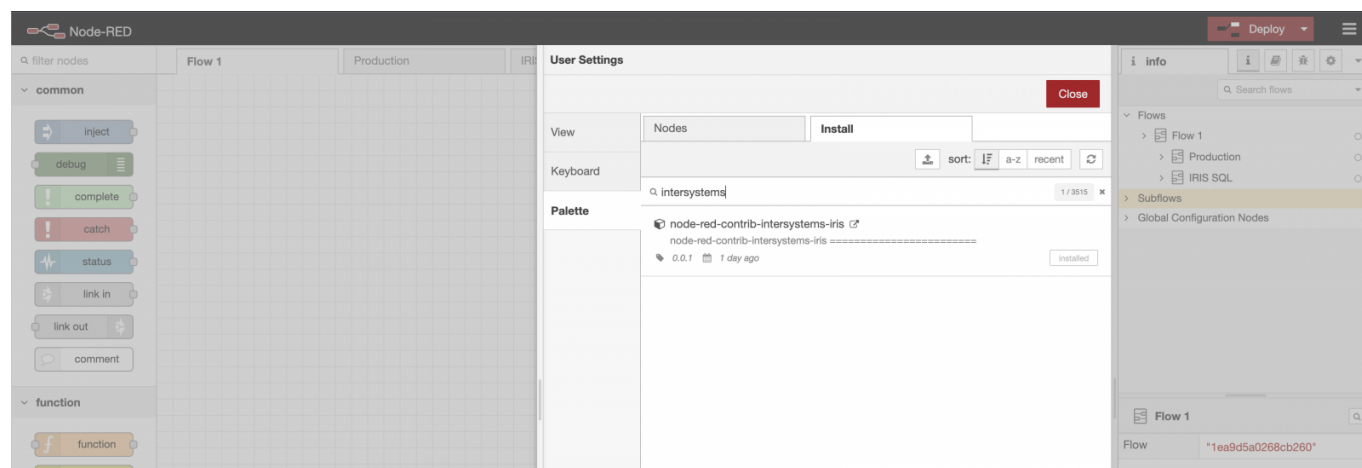
After the start of the container, it will be available by link <http://localhost:1880/>

Let's have a look at how it looks like. The field in the middle is used to construct flows by drag-n-drop nodes from the left side. On the right side, some additional control over it, such as, quick access to the items, and debugging.

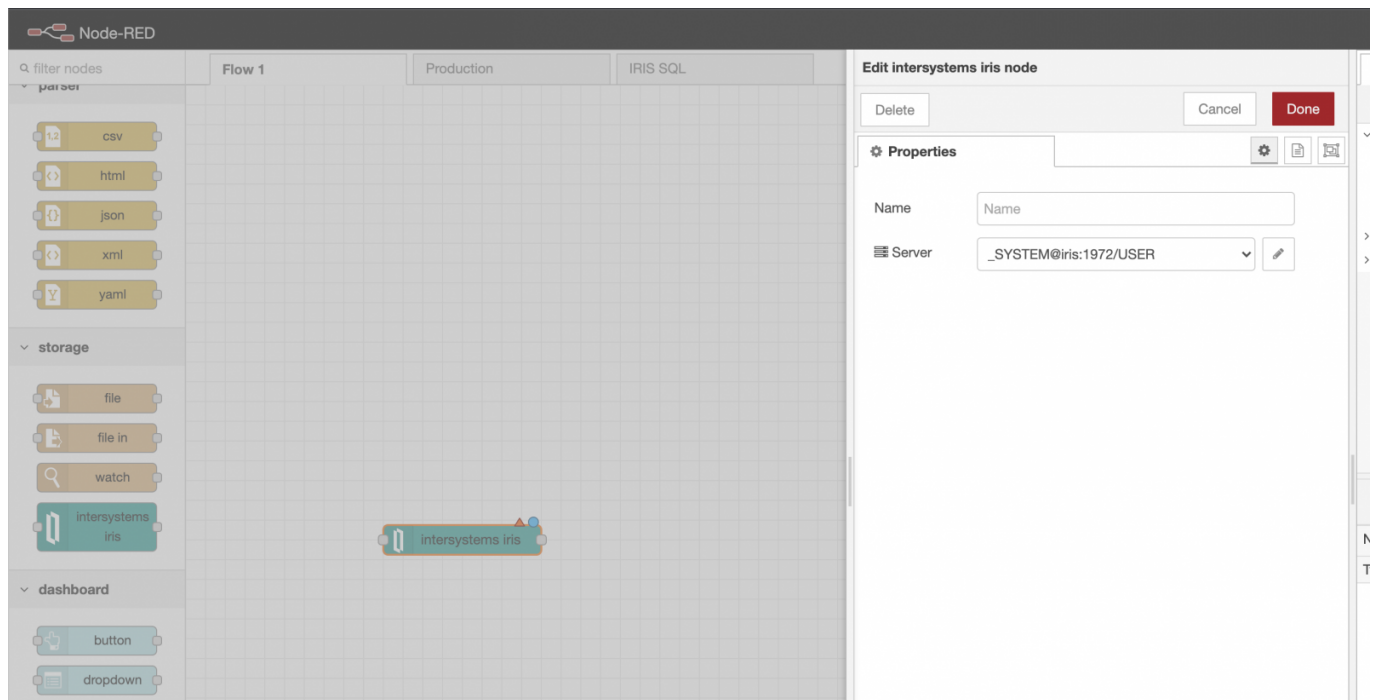


Node-RED is extendable, and we can install any new types of nodes and can find them through Palette, which can be accessed via the menu.

And we will need, to install a node, which gives access to InterSystems, just search for intersystems there and press install.



After install it will appear on the left side, in section Storage. By double click, we can open settings for the particular instance of the node.



It is possible to have many instances of InterSystems IRIS nodes there, which will use the same settings to the server, so, these settings have to be configured once.

Edit intersystems iris node > **Edit InterSystemsIRISConfig node**

Delete

Cancel

Update

⚙️ Properties

🔑 Name

\_SYSTEM@iris:1972/USER

🔗 Connection

📄 Host

▼ a\_z iris

Port

▼ 0\_9 1972

Namespace

▼ a\_z USER

Username

▼ a\_z \_SYSTEM

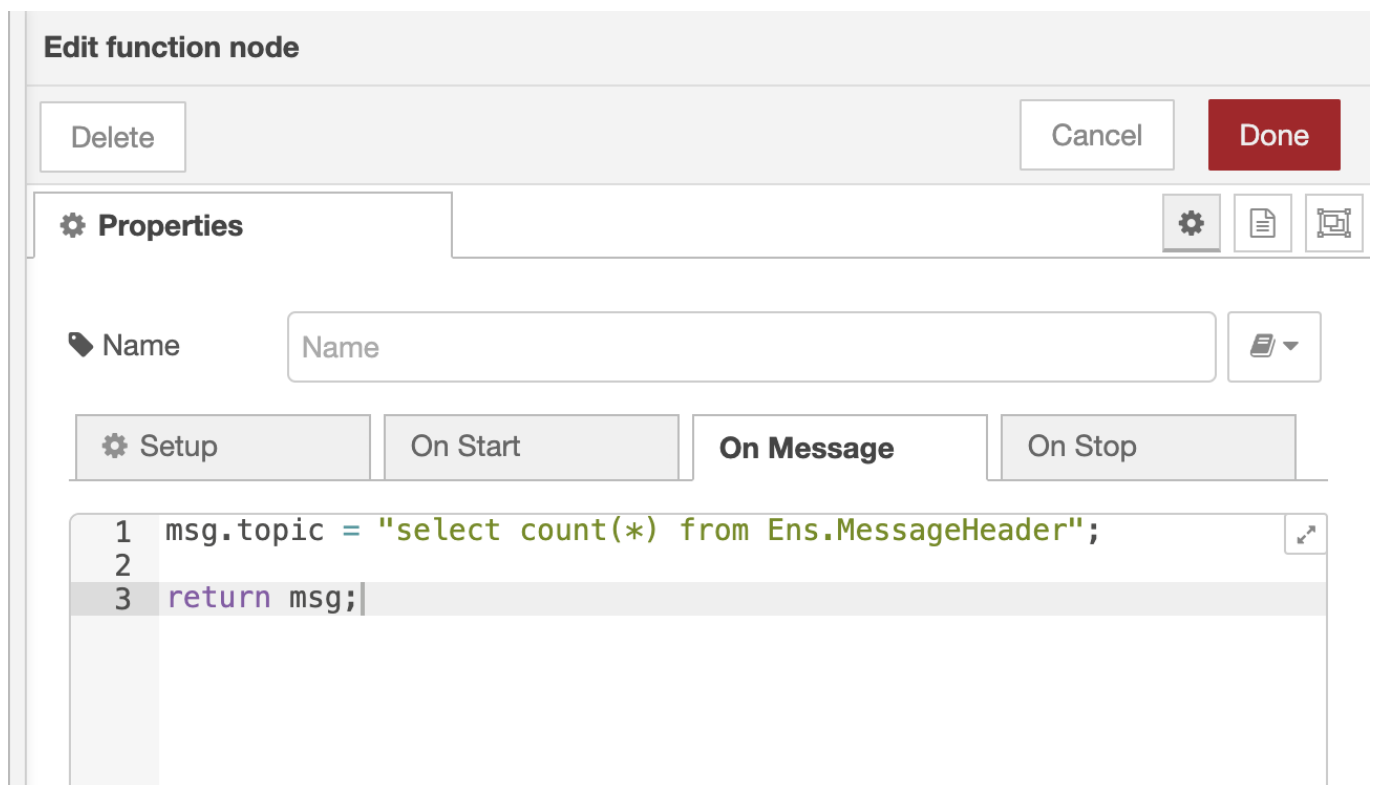
🔒 Password

▼ a\_z ...

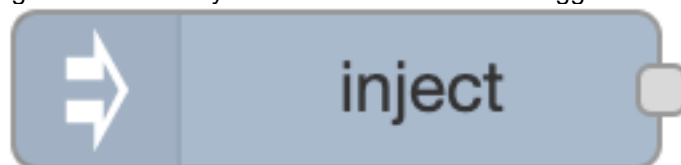
We have the node, which will do the work, then we need some node, which will provide the task, what exactly to do. InterSystems IRIS node accepts SQL Query passed as a message in property topic. And the simplest way to produce this message in Node-RED is with the node function



. These types of nodes, executes JavaScript code, it gets the messages in msg variable, can modify it and return, so, it will be sent to the next node. For the demo, we don't need anything in input, we just generate a simple message with a simple SQL query in the topic.

The image shows the 'Edit function node' interface in Node-RED. At the top, there's a title bar 'Edit function node' with 'Delete', 'Cancel', and 'Done' buttons. Below is a 'Properties' section with a 'Name' field. Underneath are tabs for 'Setup', 'On Start', 'On Message' (which is selected), and 'On Stop'. The 'On Message' tab shows a JavaScript code editor with three lines: '1 msg.topic = "select count(\*) from Ens.MessageHeader";', '2', and '3 return msg;'. There are also icons for settings, a document, and a refresh button.

It's not enough, to get it working. We need a way to send an event that will trigger the chain of nodes. In this simple case, we can use node inject



. The settings for this type of nodes allows generating a message as well, and most importantly it can be configured to inject these messages with some interval

☐ Inject once after  seconds, then

🔄 Repeat

interval



every

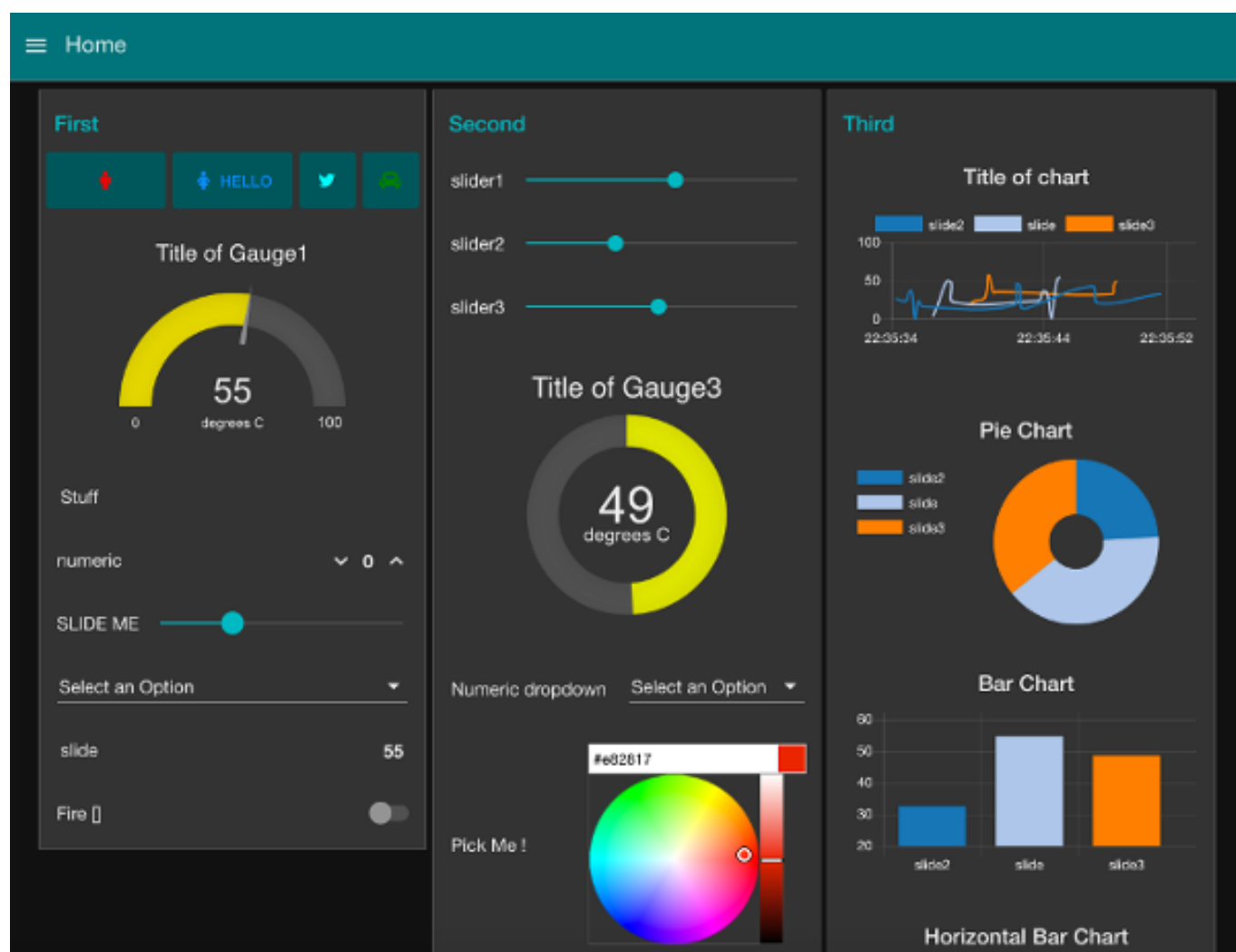
1



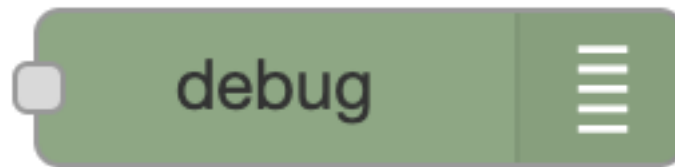
seconds



The last thing left is visualization. Node-RED has another installable package named node-red-dashboard, which provides a bunch of nodes and the capability to create Material-design based dashboards

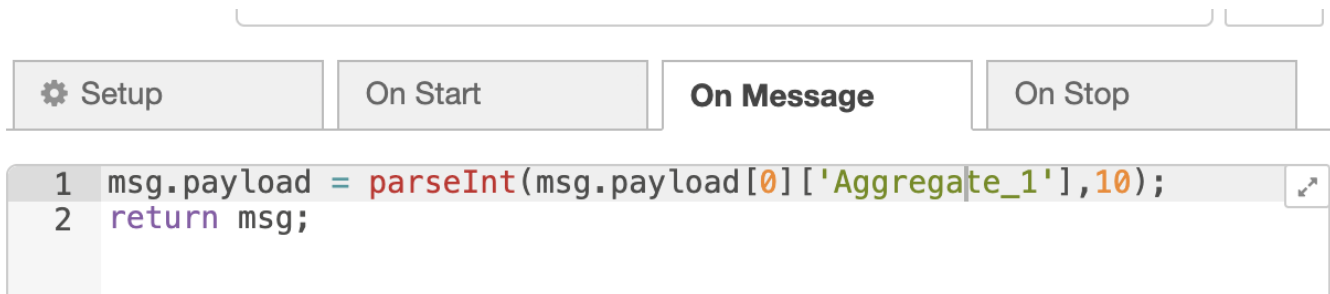


Our simple SQL query returns only one number, with the number of messages sent. So, we may use a Gauge widget or just a Text widget. Both of these widgets expects some number in msg.payload but intersystems IRIS node returns more than just a number, and we need one more function node, which will help to convert the query result to the expected format.

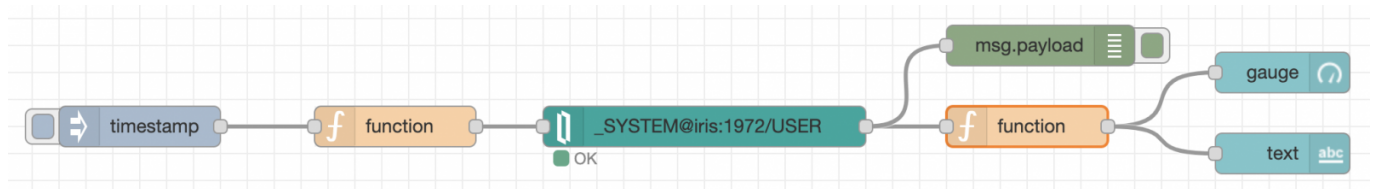


understanding how messages look like.

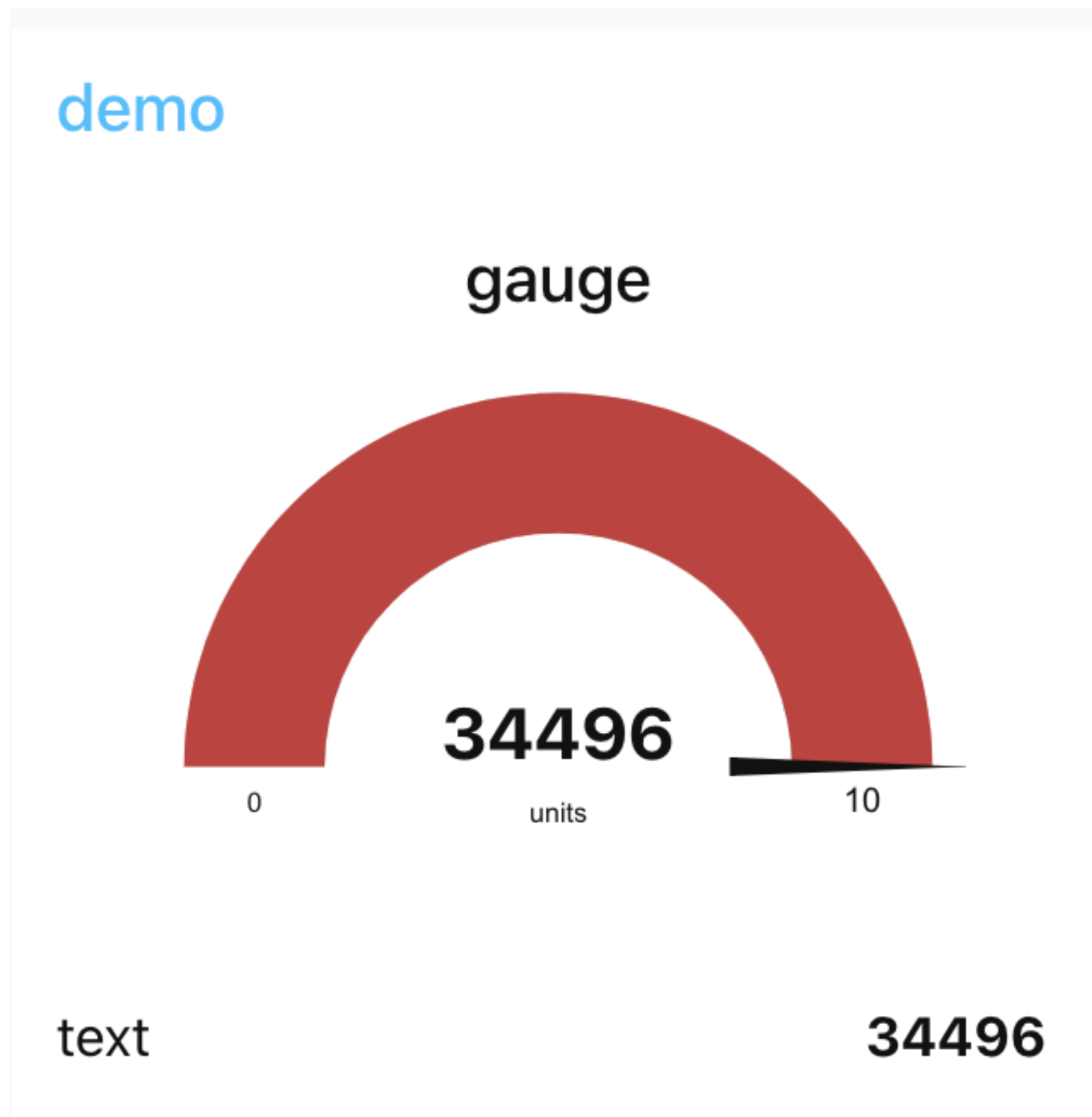
Debug node, may help in



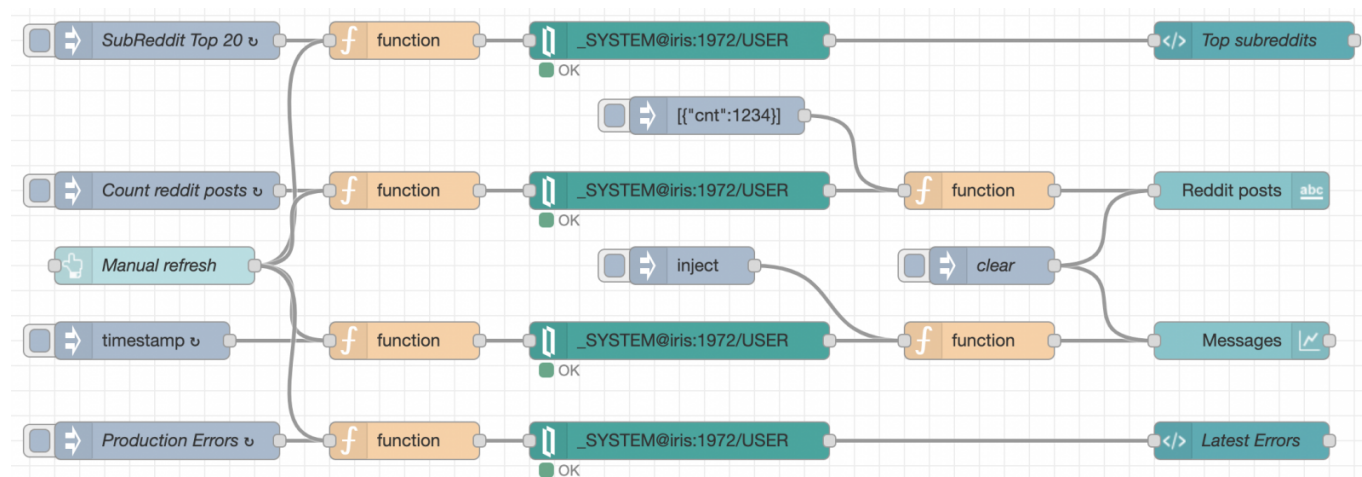
And finally, we can link all the nodes together, deploy them and test it with the small button on Inject node, to manually inject the message.



The dashboard can be found in the menu. And we will see there something like this



The flow for the first picture above is a bit bigger.



This demo is available in the [GitHub repo](#), and can be run with docker-compose.

And please vote for the project on the [Contest](#)

[#Interoperability](#) [#InterSystems IRIS](#)

[Check the related application on InterSystems Open Exchange](#)

---

Source URL: <https://community.intersystems.com/post/monitor-production-node-red>