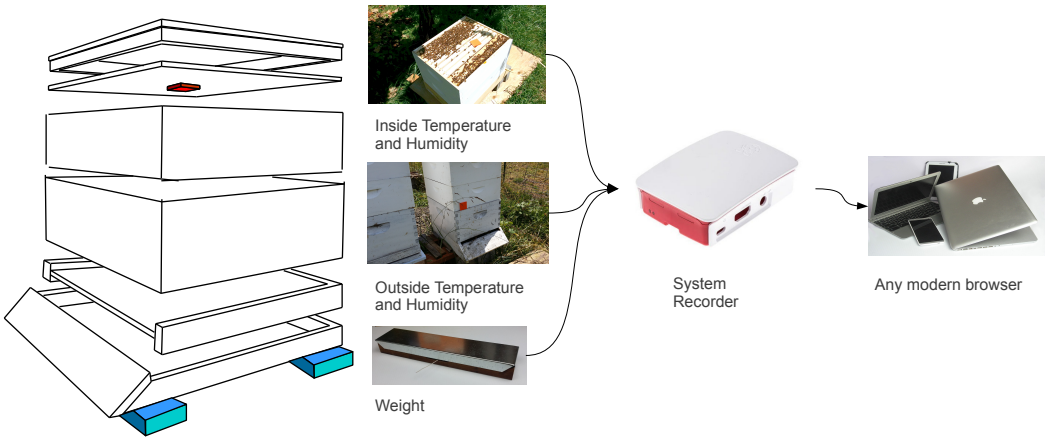
Building Custom Displays
for the Bee Certain
System Recorder

Bob Barter

Bee Certain, LLC
July 11, 2016

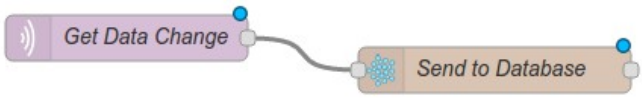
This is an advanced topic intended for those beekeepers that would like to customize the display of beehive data collected by the Bee Certain System Recorder. The Bee Certain Wireless Hive Health Monitoring System consists of three major parts. Environmental Sensors; a System Recorder; and a User Interface. The Environmental Sensors send temperature, humidity, and weight data to the System Recorder. The System Recorder collects and stores the data for multiple years and displays that data in a User Interface. The User Interface is viewed in any modern Web Browser that connects to the System Recorder through WiFi.



It is important to note that the system operates "out of the box" without the need for any configuration. All the beekeeper has to do is throw a sensor into a hive and plug in the System Recorder. No setup is required. However, the System Recorder can be customized and extended in multiple ways.

The purpose of this document is to give examples of how to extend the capabilities of the Bee Certain Monitoring System.

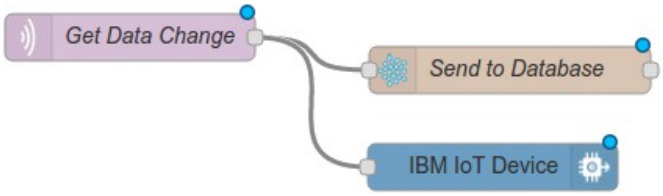
Data that comes into the System Recorder via the radio linked sensors is Published as an MQTT[1] packet to a MQTT Broker within the System Recorder. The MQTT Broker acts a the central distribution hub for all sensor data. As an example a data recording process Subscribes to MQTT data changes. When a change occurs, the process sends the new data to the System recorder's database.



The System Recorder is intended to operate standalone without the need to connect to the Internet. This is important because beehives are often kept in remote areas that do not have ready access to the

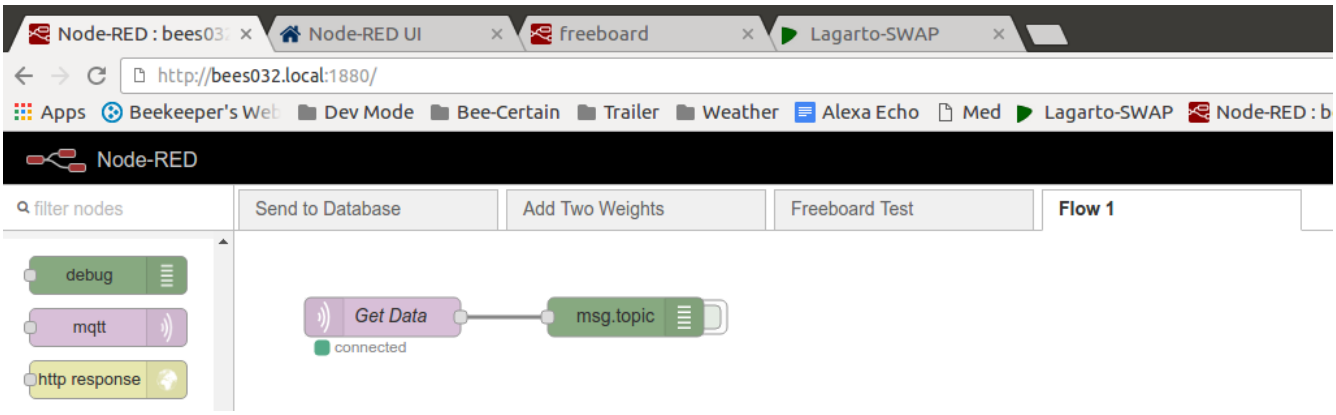
1 https://en.wikipedia.org/wiki/MQTT

Internet. However, if the System Recorder can be connected to the Internet, then it is possible to send data "to the Cloud". The System Recorder comes per-configured to send data to several Cloud Services, but can also be extended to additional services as well. Here is an example of adding Cloud connectivity to the above data flow. In this case sending messages to the IBM Watson Internet of Things Platform.

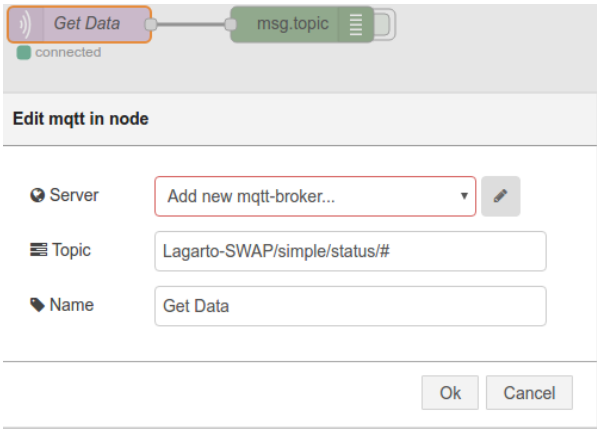


For the purpose of this paper, we will build a simple user interface that displays temperature and humidity values on a dashboard. At the risk of repeating what has already been said, all of this capability exists in the System Recorder, but none of it is required to use the System Recorder out of the box.

We are using a System Recorder with the name bees032.local, so all of the screenshots will show that URL. Your System Recorder will most likely have a different name/URL, so please adjust accordingly. First go to http://bees032.local:1880/ to get to the data flow interface. We are using Node-RED[2] for our flow programming. Create a new tab, in this case it defaults to "Flow 1", and add an MQTT node and a Debug node.

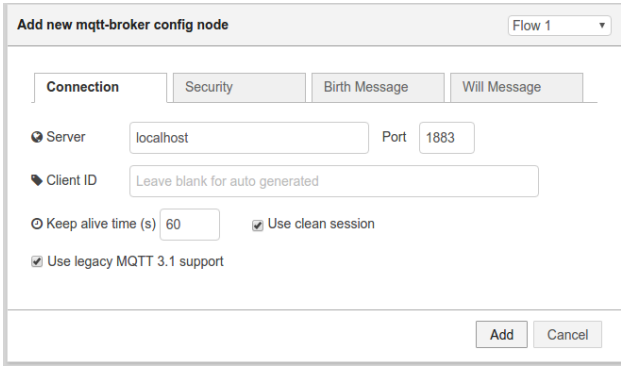


Double click on the MQTT node. The Name can be anything you like. We are using "Get Data". For Topic, enter "Lagarto-SWAP/simple/status/#", without the quotes. Next, click on the Edit icon at the far right of the Server line.

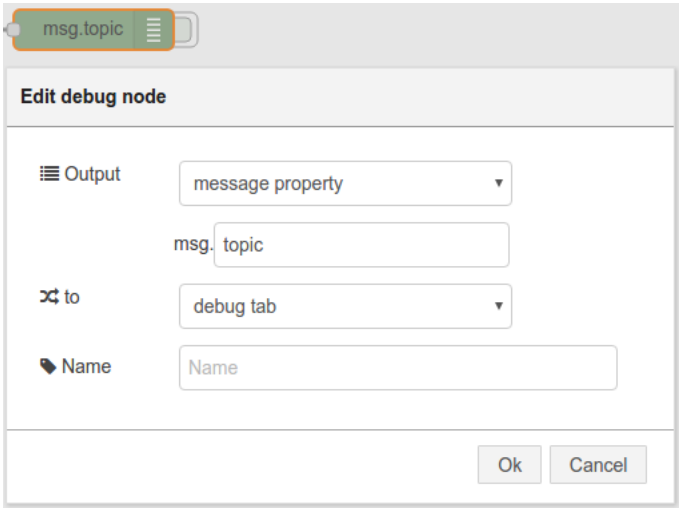


2 http://nodered.org/

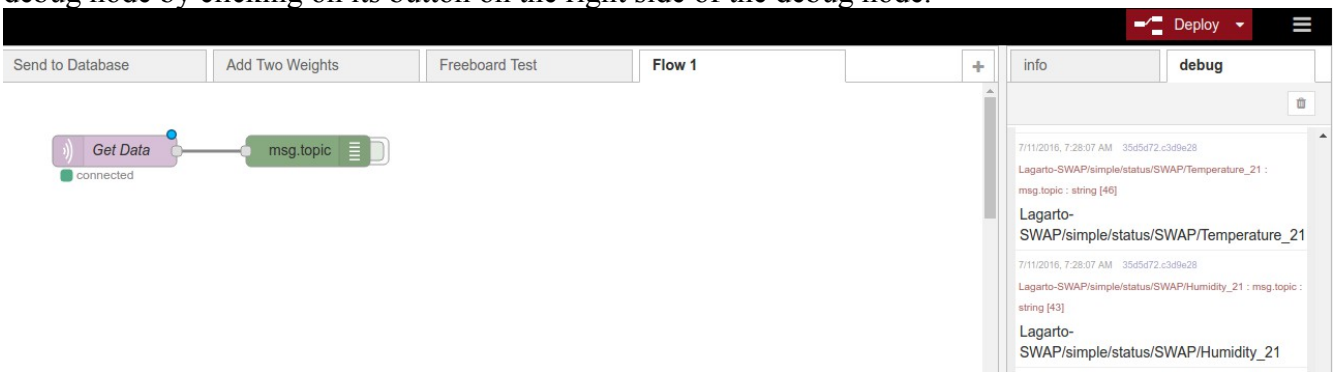
The Server dialog needs only one entry. Type "localhost" without the quotes in the Server field. Click add to go back to the Edit MQTT screen and click OK.



Now double click on the Debug node. To start, we want to see the Topics that we are receiving, so change msg.payload to msg.topic. Click OK.



Only a couple of more steps. First wire up the output from the MQTT node to the input of the Debug node. Click the Debug tab on the right. Click the Deploy button in the upper right, and turn on the debug node by clicking on its button on the right side of the debug node.

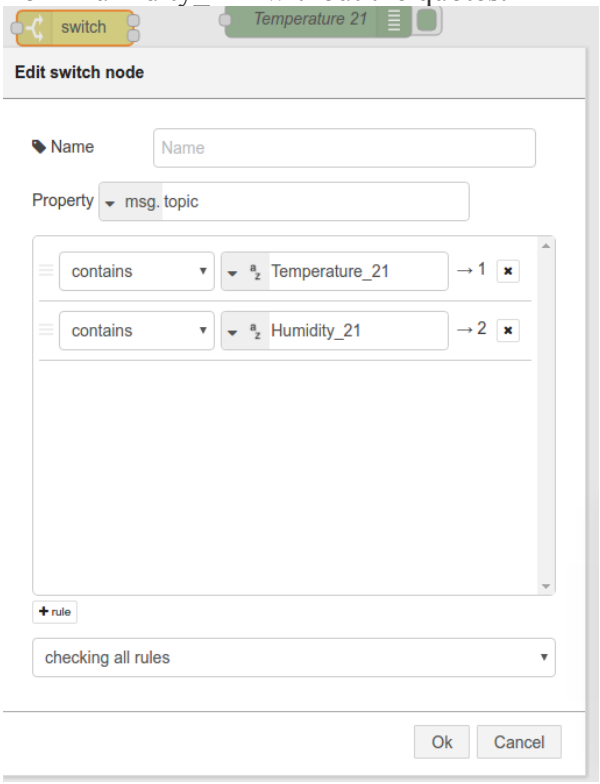


If your System Recorder is receiving data from your temperature and humidity monitors, then you should see debug messages like the ones in the above image.

Note the topics like:
Lagarto-SWAP/simple/status/SWAP/Temperature_21
Lagarto-SWAP/simple/status/SWAP/Humidity_21

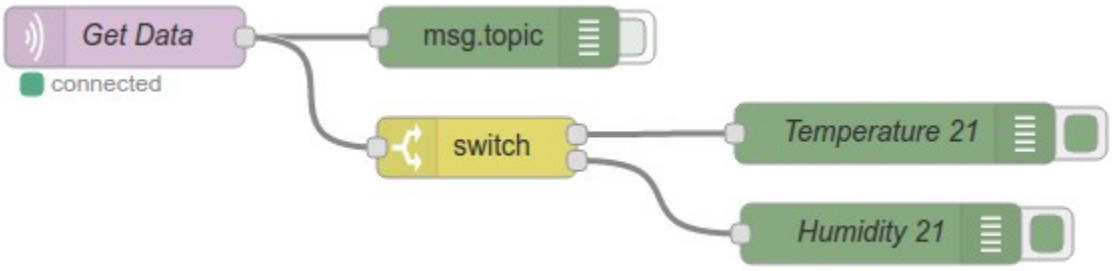
These look interesting. Add a Switch node and two more debug nodes. Next, we'll go through their configuration. Note: the names of your temperature and humidity sensors will most likely be different, so adjust accordingly.

Let's configure the Switch node to have two conditions. We read in the msg.topic and then test to see if it contains “Temperature_21” or “Humidity_21” without the quotes.

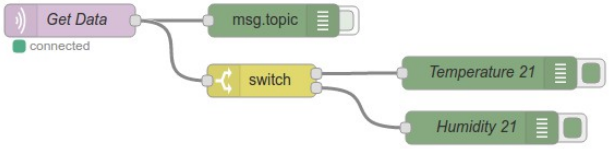


Note that two test conditions result in two outputs from the node.

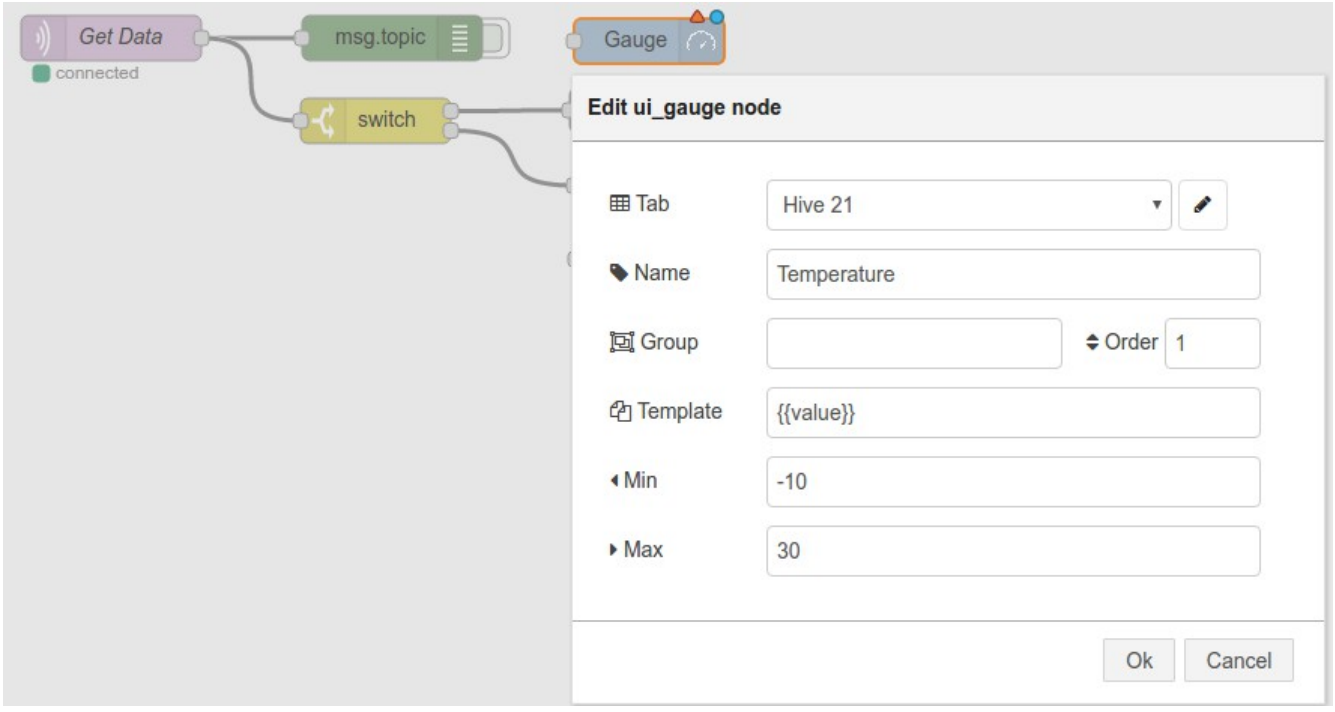
All we do to configure the Debug nodes is change their names so they are more meaningful. Next we wire up the nodes, enable them and click Deploy.



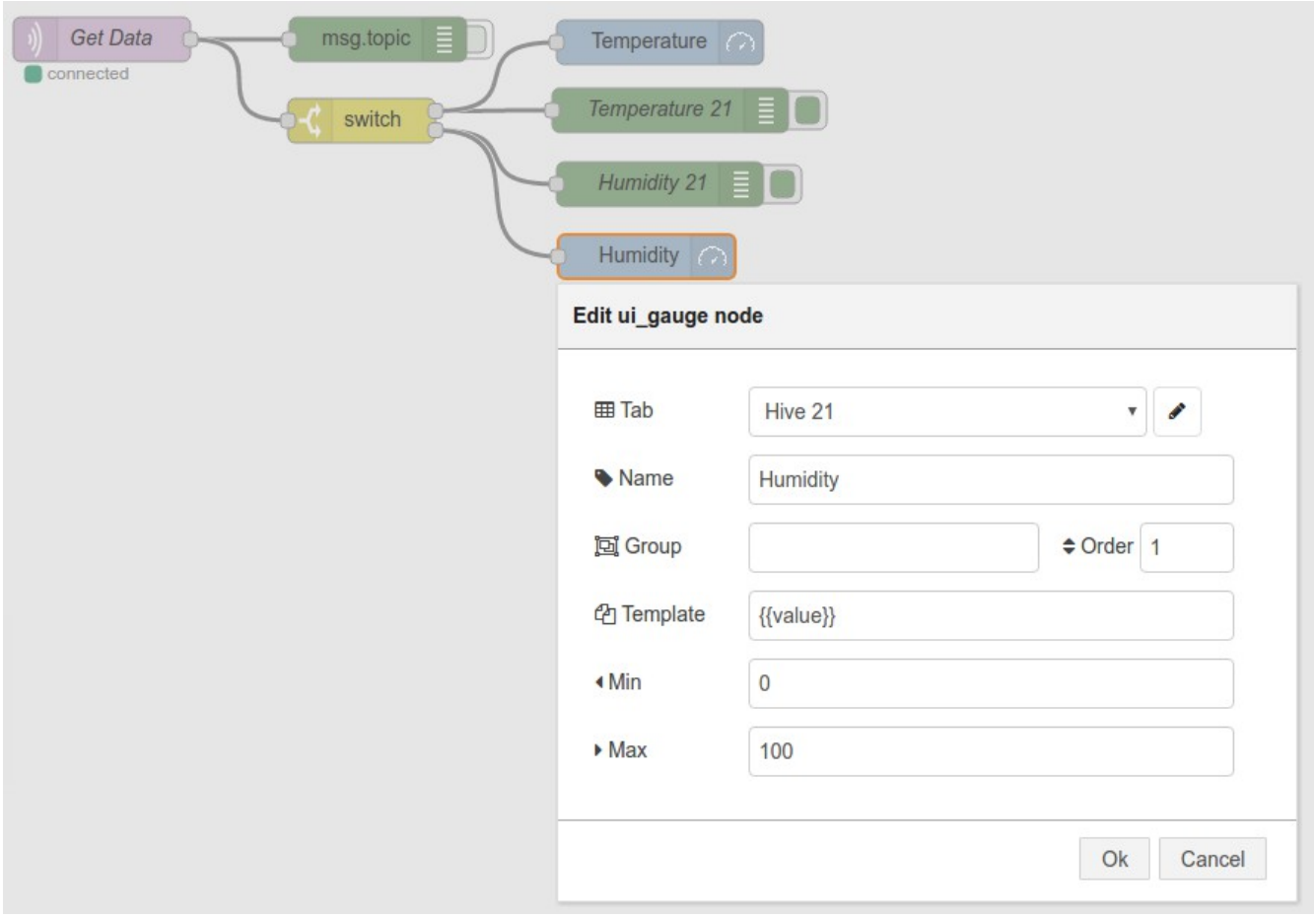
Now we begin seeing these debug messages.



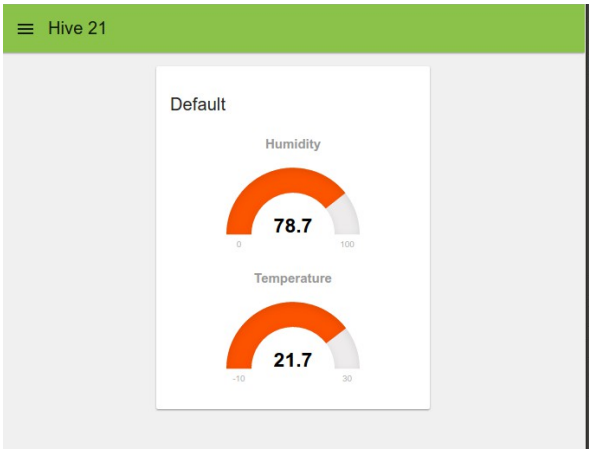
Now lets add a couple of User Interface elements. Add a gauge and double click. We will create a new tab called “Hive 21”, give the node the name “Temperature” and set its min and max values to -10 and 30.



Do the same for a humidity gauge with the same tab name “Hive 21”, give it a name Humidity with min and max values of 0 and 100.



Now hit the Deploy button and go to a new URL in a different browser window: http://bees032.local:1880/ui and find the tab “Hive 21”.



These are just a few of the things that we can due with the flow programming model in Node-RED. Stay tuned for more examples. Has your queen sent a Tweet lately? How about an email whenever the humidity in a hive gets over 80%?

Bee Certain

For more information:
http://bee-certain.com/