Eastland Farms

IoT Weather

Demo

# Setup video

1. Watch the setup video and create the demo environment using the IoT Starter boilerplate. You will also get a feel for the construction of the overall demo.

**\*NOTE\*** the flows have changed a little since the video was created

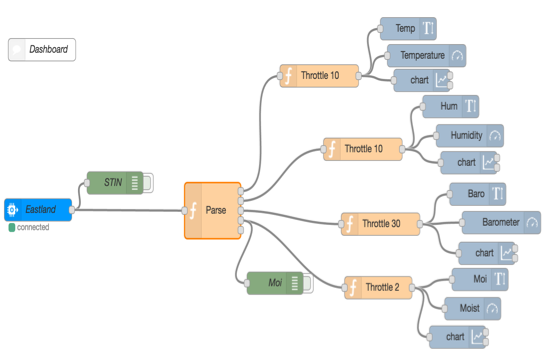
1. Don’t forget to register your Sensor Tag to the IoT Platform.

Make note of the **organization**, **device id**, **user id** and **password**

1. The TI application will need to be updated with the registered device information in the Cloud configuration section.

# Dashboard UI

The Dashboard flow has two flows, the top flow takes the input from the Sensor Tag and populates the **Node-RED-contrib-ui** widgets.

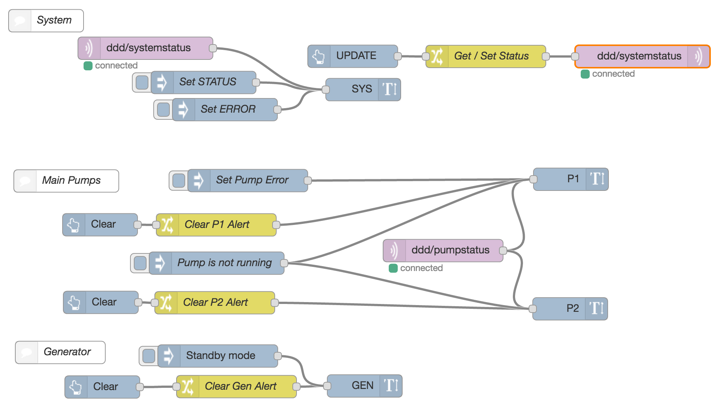


This flow parses the temperature, humidity, barometric pressure, and the light reading from the Sensor Tag payload. These values are pushed into the contrib-ui widgets. The light reading is forwarded as the moisture content. See the **‘Parse’** node.

The four **Throttle** nodes are used to limit the message rate into the UI widgets. The Sensor Tag publishes a message a second, these nodes throttle that back so the dashboard doesn’t get overwhelmed.

A key function is to store the sensor data values in global values. These variables are then read from the global context by the Weather flow in the ‘**Decision Engine**’node.

The bottom area of the flow tab handles the system status.



This flow uses the MQTT input and output nodes (purple) to update the system status and the pump status. The rest of the ‘backend’ system is simulated using several Inject nodes. The flow also demonstrates the use of the contrib-ui Button to clear several fields.

1. Import the **Dashboard UI** flow
2. The url for the dashboard is the base application url + ‘/ui’

e.g.

<http://weather-ddd.mybluemix.net/ui>

**\*NOTE\*** the Node-RED dev environment is at:

[http://weather-ddd.mybluemix.net/red/#](http://weather-ddd.mybluemix.net/red/)

1. IoT Input node

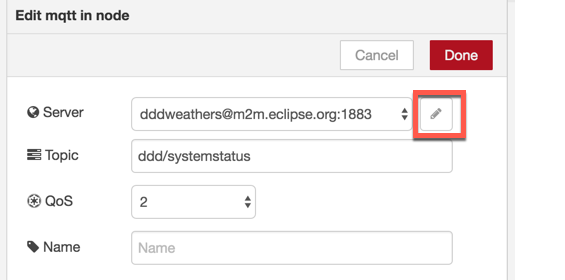
This is the only node that needs to be changed in the top flow.

Change the device id to match the one you registered to the IoT Platform.

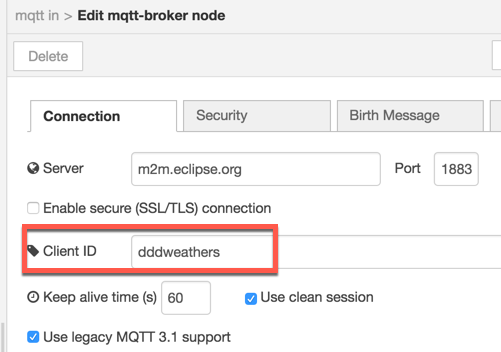
1. MQTT nodes

Change the server configuration to use a unique client id.

1. Open an MQTT node and edit the server config by clicking the pencil.



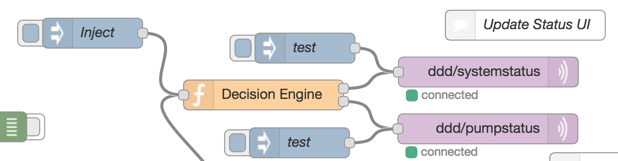
1. Change the client id to something unique using your initials. **Don’t** use the existing ‘dddweathers’



1. You only need to change the server config in one MQTT node per flow tab as it is shared by all the MQTT nodes on that tab.
2. The server being used is “**m2m.eclipse.org**”. This is a sandbox server that is publicly available and has the security turned off.
3. Save the configuration and redeploy the flow.

# Weather flow

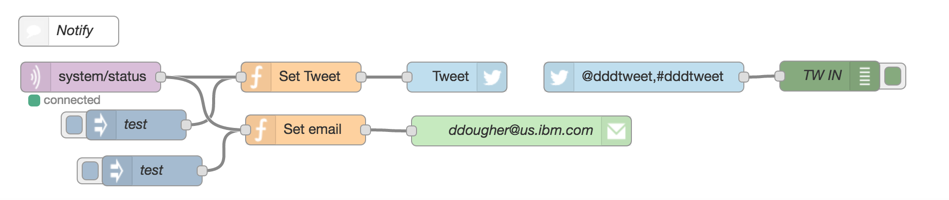
This flow has several sections we will break down. The first section is:



This is the ‘**Decision Engine**’. This node reads the sensor values from the global context, that were set by the UI Dashboard flow. It has a simple algorithm to look at the temperature and moisture content. It publishes the ‘decision’ to the system status on the dashboard.

**\*NOTE\*** the Inject node can be used to drive this part of the flow when doing the sensor only part of the demo. See the ‘**Running the demo**’ section.

The next section is the ‘**Notify**’ section at the bottom:



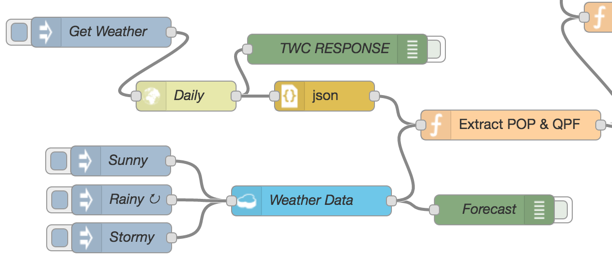
This flow subscribes to the system status published in the Decision Engine flow.

You can use this during any part of the demo to make the demo more ‘exciting’ by notifying an audience member via Twitter or Email.

**\*NOTE\*** you must have a valid Twitter or Email account to use these nodes

Do **not** use the existing credentials if present.

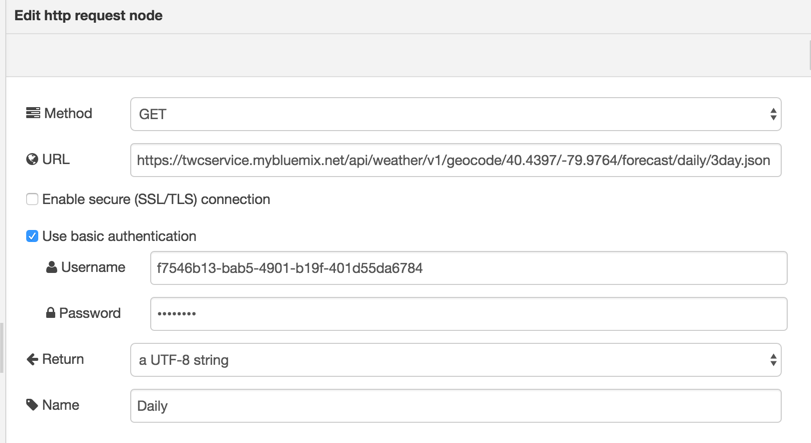
The next flow section gets the Weather forecast:



The node labeled ‘**Daily**’ is an HTTP Request node that calls the Weather api to get the daily forecast. The http response is converted to JSON and the ‘Extract’ node parses out the two fields we are interested in, the ‘**POP**’ and **QCF**’, which are the **chance of precipitation** and the **expected rainfall** respectively.

The pop and qpf are then passed to the Decision Engine to be used along with the sensor temperature and moisture content.

This is the HTTP Request node configuration:



The username and password come from the **VCAPS** variable for the Weather Data service you instantiated earlier. You can also get this information by clicking on the ‘**Show Credentials**’ for your weather service in the Bluemix dashboard.

The URL for the daily 3 day forecast is:

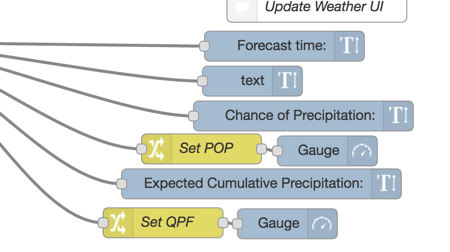
<https://twcservice.mybluemix.net/api/weather/v1/geocode/40.4397/-79.9764/forecast/daily/3day.json>

The geocode passed is for Pittsburgh,PA.

The node labeled ‘Weather Data’ is a Cloudant database where I have stored three forecasts to call as needed. They are retrieved using the inject nodes labeled ‘Sunny, ‘Rainy’ and ‘Stormy’. See the ‘**Saving forecasts in Cloudant**’ section.

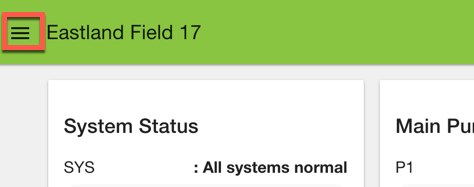
**\*NOTE\*** You can use the HTTP Request node to examine what a forecast looks like in its entirety, but when I run the demo, I use the inject nodes to drive the saved forecasts at regular intervals into the Decision Engine. See the ‘**Running the demo**’ section.

The last part of the flow is for showing the forecast on another page in the dashboard UI.

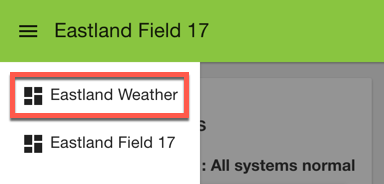


This is for seeing the forecast summary, pop and qpf in a more readable format than the Node-RED debug pane during the demo. See the ‘**Running the demo**’ section.

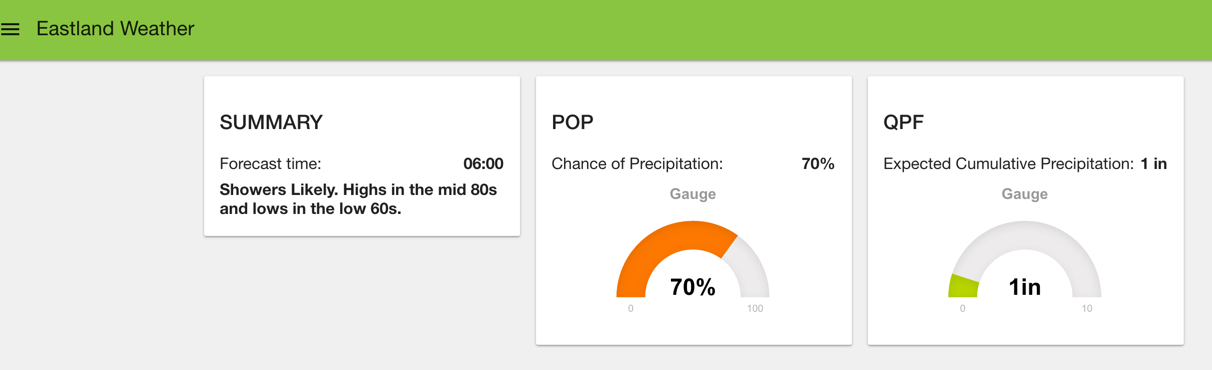
This page is accessed from the Dashboard by clicking the information icon in the upper left hand corner:



And selecting “Eastland Weather”.



The page will look like this:



1. Import the Weather flow
2. MQTT Nodes (the purple ones)

Update the client id in the server configuration as before, but you must use a unique id for each flow tab on the canvas.

e.g.

If you used **‘abcweather1’** in the Dashboard flow use **‘abcweather2’** in the Weather flow

# Running the demo

The demo has two main parts, first the sensor only and then the sensor and weather data part. The combined part has three sub parts – the Sunny, Rainy and Stormy forecasts.

**\*Note\*** remember the demo changes the Light Lux reading to Moisture Content. So when you manipulate the light sensor you are really adjusting the moisture content.

**Sensors only:**

The Dashboard flow processes the sensor data and stores in the global context. The Weather flow reads from this context.

To drive the Decision Engine in the Weather flow, I set the node labeled ‘Inject’ to an interval of 5 secs. This way I can focus on the changes on the Dashboard as I manipulate the Sensor Tag to adjust the moisture content.

1. Set Inject interval to 5 secs. Redeploy flow.
2. Show UI Dashboard while manipulating the Sensor Tag

**Sensor and Weather**

I used the same strategy here, set the Sunny, Rainy and Stormy inject nodes to intervals of 5 secs, **one at a time**, or you will confuse the Decision Engine.

e.g.

1. Set Sunny inject interval to 5 secs. Redeploy flow.
2. Show Weather Dashboard to see the forecast.
3. Show UI Dashboard while manipulating the Sensor Tag
4. Repeat for Stormy and Rainy forecasts.

**\*Note\*** See the ‘**Saving forecasts in Cloudant**’ section below.

# Saving Forecasts in Cloudant

The ‘**Save Forecasts**’ flow will call the daily weather service for a particular region based on geocode and store the forecast in a Cloudant database. Once you have saved forecasts, you can drive the demo in a predictable manner.

1. Create a Cloudant database called ‘**weatherdaily**’ (all lowercase)
2. Import the ‘**Save Forecasts**’ flow
3. Change the database on the Cloudant node to ‘**weatherdaily**’
4. HTTP Request node labeled ‘Daily
5. Check ‘use basic authentication’
6. Add your weather service credentials – user id and password

I saved off 3 forecasts over the course of several weeks:

1. Sunny, no rain
2. Rainy with a little rain < 1 inch
3. Stormy with a lot of rain > 4 inch

You could be clever with the geocode and find three appropriate forecasts in a matter of minutes.

1. Look at a weather map and locating areas of the desired weather.
2. Use a tool such as Google Earth, to determine the geocode longitude and latitude for that area. The structure of the geo code is:

…/v1/geocode/latitude/longitude/…

1. Insert the geocode into Weather api URL in the ‘**Daily**’ HTTP Request node.

Or you can use the postal code for a location if you have it like this:

…/v1/location/postalCode/….