Temperature Sensor using Intel Edison board

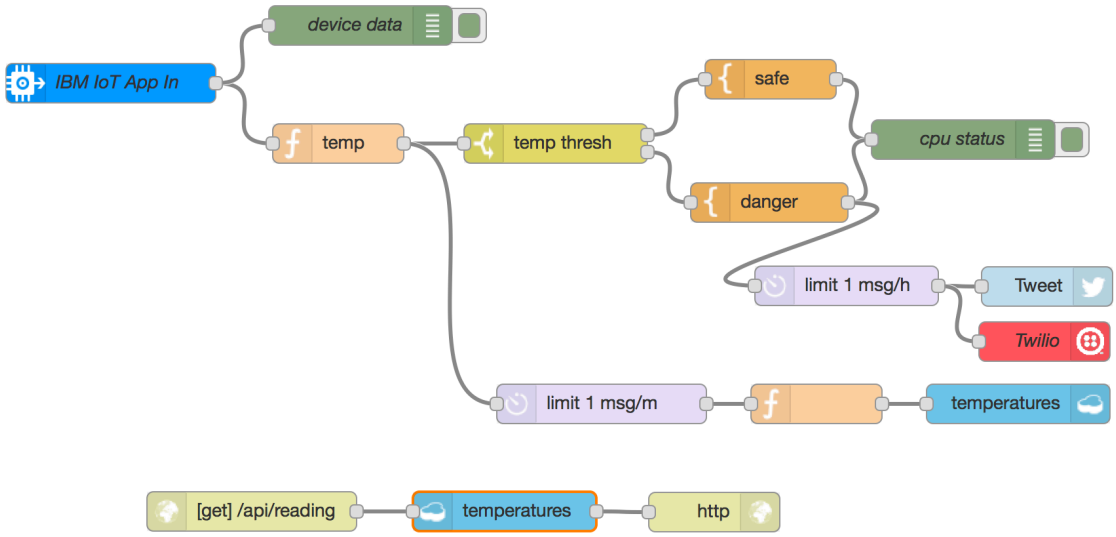
Hands-On Lab

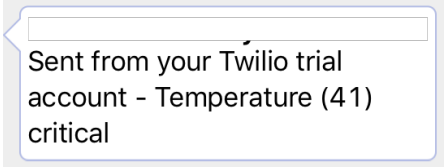
Author: JeanCarl Bisson | jbisson@us.ibm.com | @dothewww

Table of Contents

Topics covered: Intel Edison, Grove sensors, Node-RED, Internet of Things, Cloudant NoSQL Database, Twitter, and Twilio.

In this lab, we will use Node-RED on an Intel Edison board (with a Grove Starter kit) along with several services available in IBM Bluemix. Node-RED, running on the Intel Edison board, will read the temperature value from a Grove temperature sensor. The temperature will be sent via IBM's Internet of Things Foundation service to a Node-RED application hosted on IBM Bluemix. If the temperature reaches 40°F, we'll send a tweet via Twitter, and a text message via Twilio. We'll store the temperature in a Cloudant NoSQL database so we can track patterns. Finally, we'll create an HTTP endpoint that exposes the historical temperatures that third-party applications could consume and perform analysis or other fun stuff.



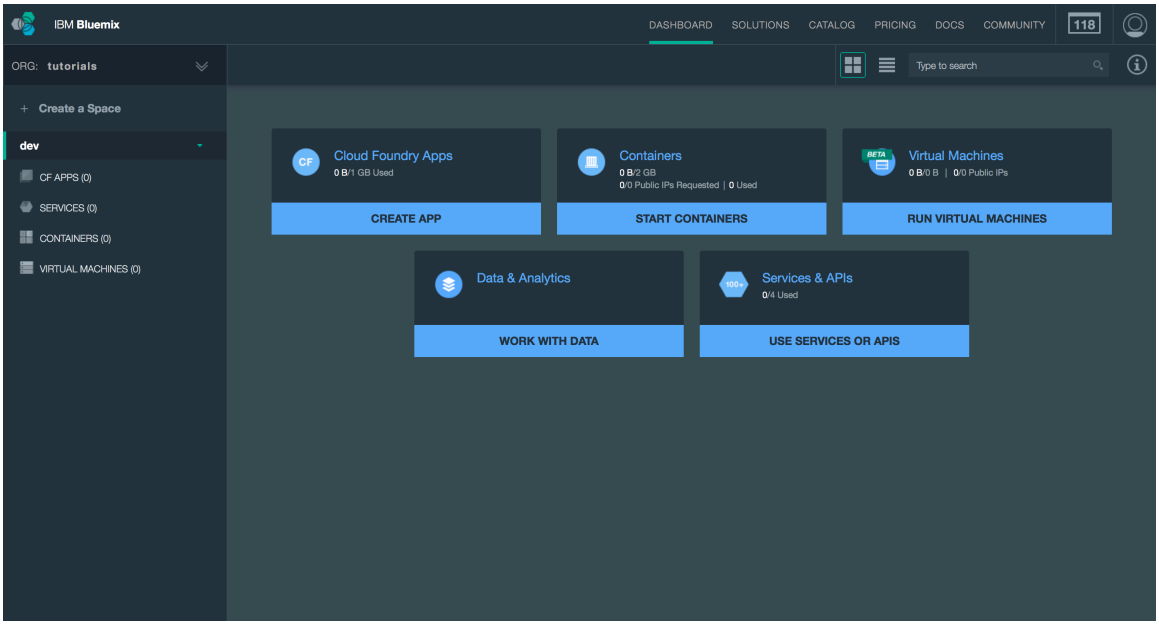



```
[{"_id":"11d0f324f4fe069825b2170198b48a04","_rev":"1-32b465fb952bea4f5196a9e654a1514e","time":1453176413834,"temp":21},
{"_id":"11d0f324f4fe069825b2170198d178bf","_rev":"1-b61bfff2e7bbf5638618aa7671013b53","time":1453176654950,"temp":21},
{"_id":"16786b564a223659b3ad69dfebae992e","_rev":"1-a826af50277e9849845ff0eed6e58c8b","time":1453176353627,"temp":20},
{"_id":"56e2bee50c7bbe93a22fab349438ab99","_rev":"1-6fa0f11dc0aa799bb4b48fc15af62dbf","time":1453176895907,"temp":20},
{"_id":"75ffb47b452f7ccc5c56aa0cdff73628","_rev":"1-1fd765692df1b47fd5b72c99afb72f9b","time":1453176293554,"temp":21},
{"_id":"c3516c0143f46eb63a054f743bb8ef66","_rev":"1-655f173446fcaccd5d7b9992d0e9f87c","time":1453176534279,"temp":20},
{"_id":"cc20392d54dd3b131646e7db68dd78fc","_rev":"1-c10c4410806e3922c95bd7b44fbf55fd","time":1453176474098,"temp":20},
{"_id":"ef1fa6d54f3e16c0e6b763cfd105f43a","_rev":"1-0aab58b8d693dcaac314f86e0a926866","time":1453176835565,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c11fffa9","_rev":"1-961cdac2e2fd0877480670dee961c94c","time":1453176594442,"temp":21},
{"_id":"f8c256cc283e0ab5e72420d9c130f876","_rev":"1-852900329df8aace34f9daaac6135002","time":1453176715108,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c1356c14","_rev":"1-03ecf88ac6c3c38609232e16c13b85ef","time":1453176775474,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c146897f","_rev":"1-6f69e480c133d5990d55257fbddb9fbb","time":1453176918219,"temp":21}]
```

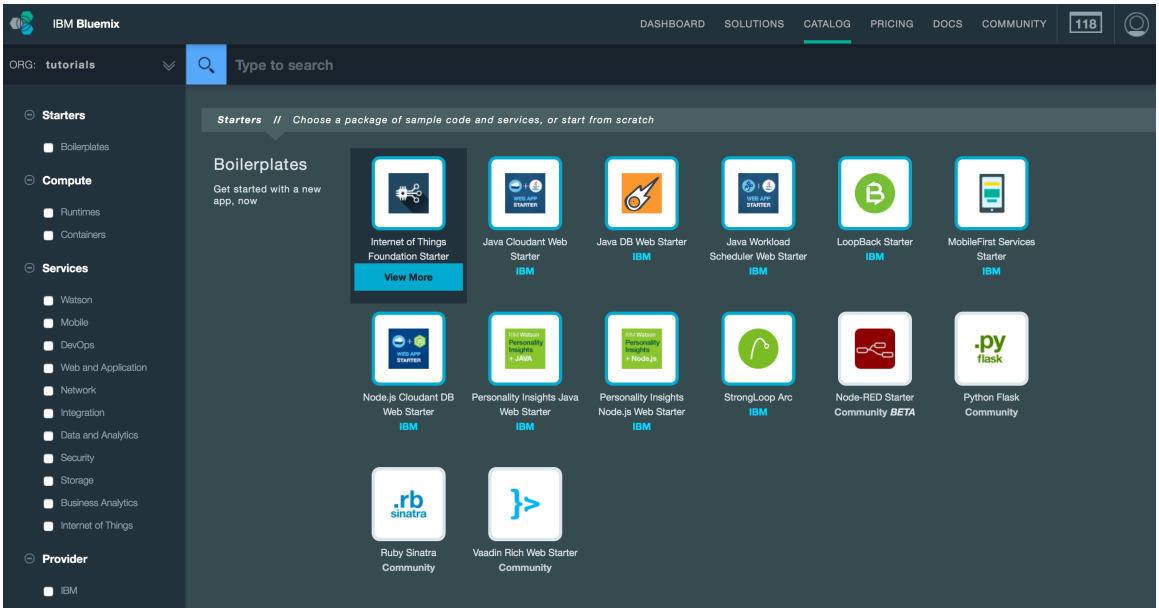
Creating an IoT application in IBM Bluemix

In this section, we will create an Internet of Things boilerplate application and use it to receive temperature values from a simulated IoT sensor.

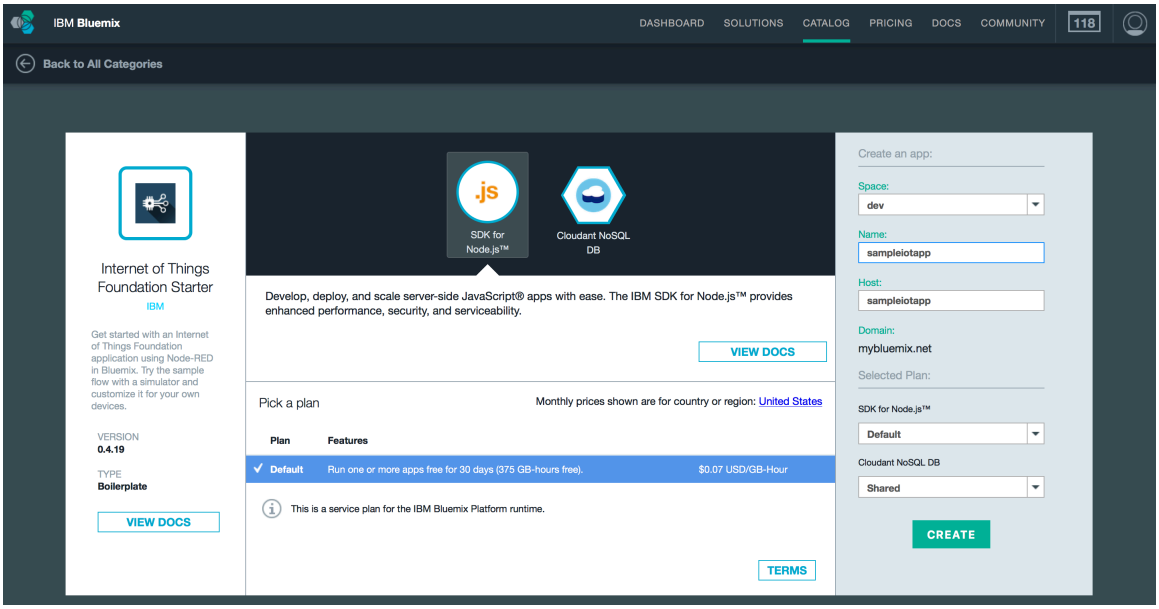
1. Open a web browser and go to your IBM Bluemix dashboard, `http://bluemix.net`. This is the IBM Bluemix dashboard where you have access to creating Cloud Foundry apps, IBM Containers, Virtual Machines, and a variety of Services. We are going to create a Cloud Foundry application today. Click on **Catalog** in the top right navigation menu.



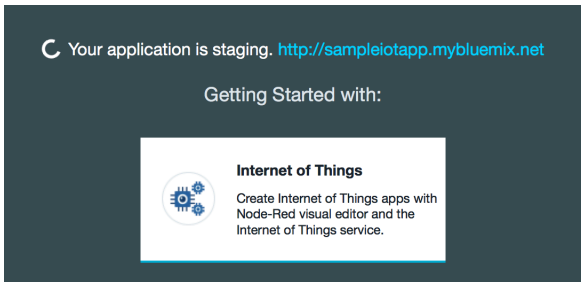
2. IBM Bluemix offers a handful of boilerplate applications that you can create and get started quickly. The Internet of Things Starter boilerplate includes Node-RED, Cloudant NoSQL database, and the SDK for Node.js. Under Boilerplates, select the **Internet of Things Foundation Starter** boilerplate.



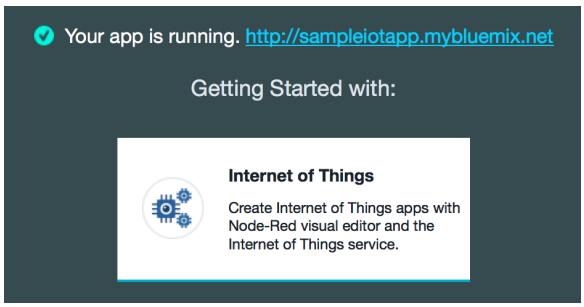
3. Pick a unique name for your application. If you choose ***myapp***, your application will be located at `http://myapp.mybluemix.net`. There can only be one "***myapp***" application registered in IBM Bluemix. You might try adding your initials in front of the host if it is already taken by someone else. Click on **Create** to create the application instance.



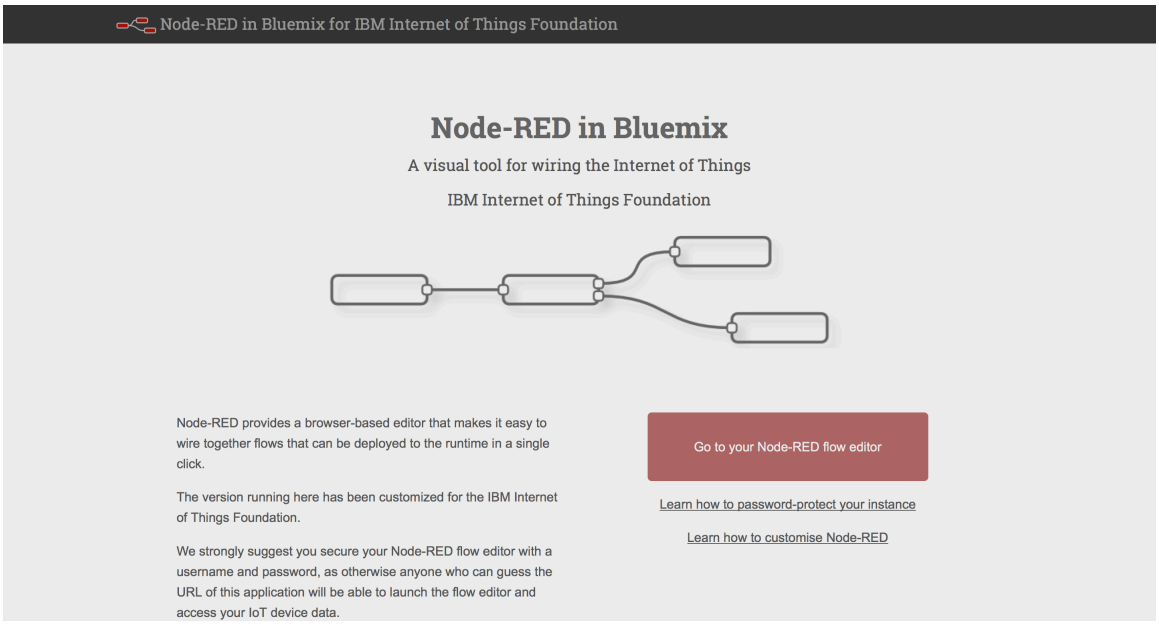
4. IBM Bluemix will create an application in your account based on the services in the boilerplate. This is called staging an application. It can take a few minutes for this process to complete.



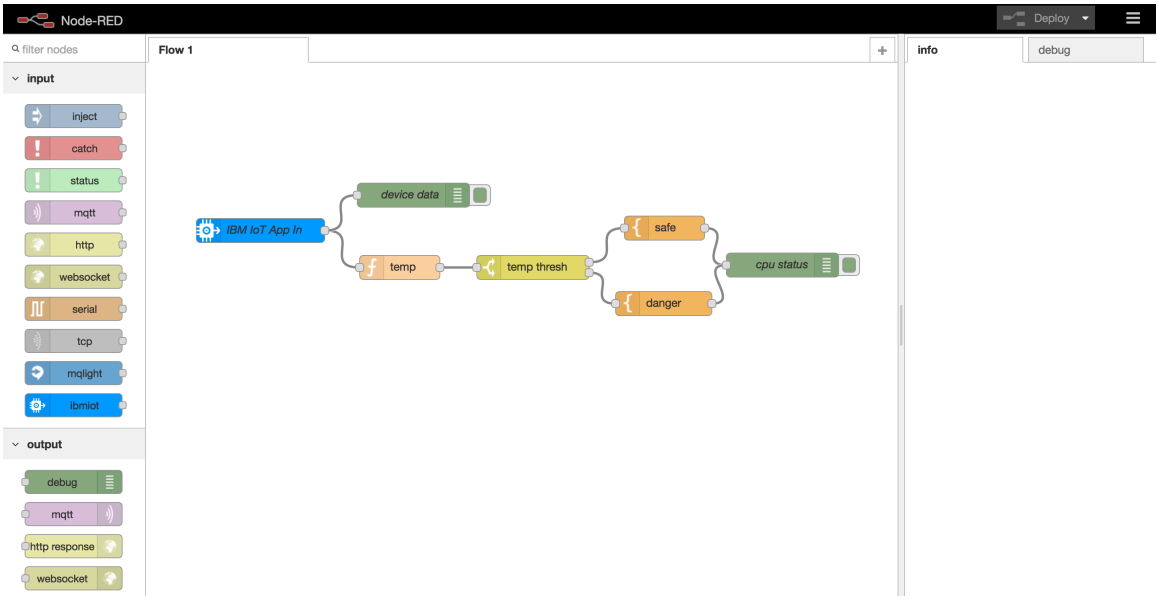
5. When the application is finished staging, the address should turn into a hyperlink. Click on the address.



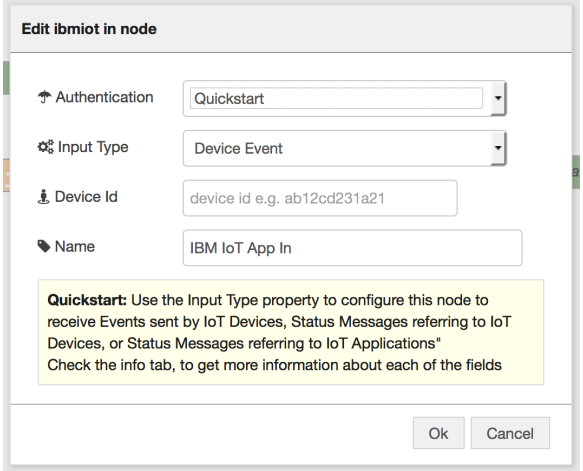
6. This is Node-RED. Node-RED is an open-sourced Node.js application that presents a graphical interface of nodes. Each node performs a defined functionality. You don't need to know how to code, just drag and drop nodes and connect them together to make an application. (Find out more at `http://nodered.org`) Click on the red button **Go to your Node-RED flow editor**.



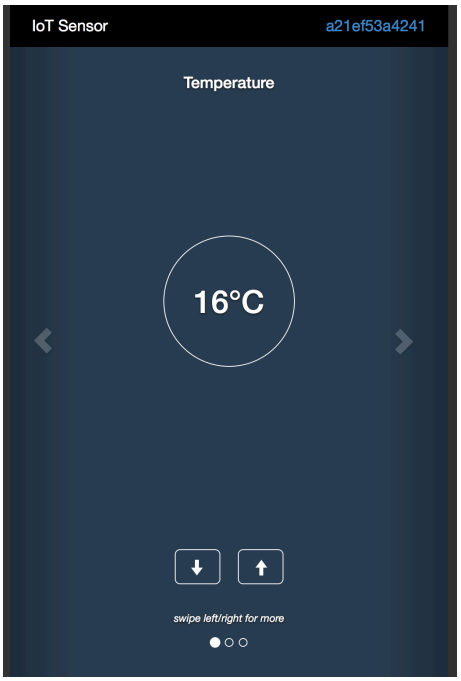
7. On the left side is a palette of nodes. Each node has a defined functionality. You can click on a node in the palette and find out what it does in the info tab on the right side of the screen. The middle pane is a canvas where nodes are placed, customized, and connected together to construct what is called a flow. A flow is executed from left to right, and is completed when the last node is reached. A flow can split off into two or more flows, for example, with a switch node. Two flows can also share a node or a flow, reusing the same logic in multiple scenarios. To customize a node already on the canvas, double click on it. Double click on the IBM IoT App In node to customize it.



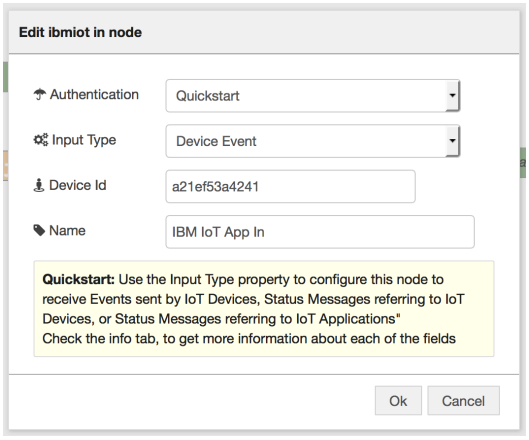
Nodes can be customized in different ways, depending on what they do. The IBM IoT node begins a flow when a message is received from a specific device. We need a Device Id of a device that is connected to the IoT Foundation. We could use a real device, or we can simulate devices.



8. To get a device ID from a simulated device, visit `http://ibm.biz/iotsensor`. In the upper right is an alphanumeric value. This simulated temperature sensor sends the temperature to the IBM Internet of Things Foundation service using this device ID.

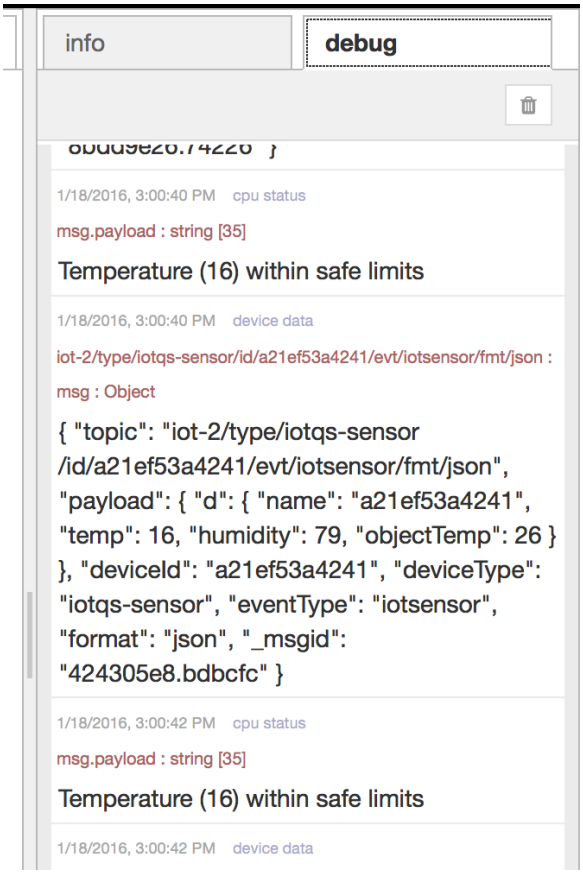


9. Copy this alphanumeric device ID into the Device ID field in the Node-RED application. Click Ok.

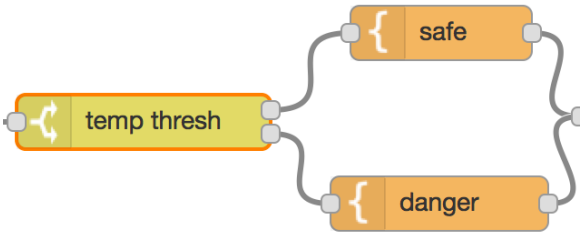


10. Click on the 

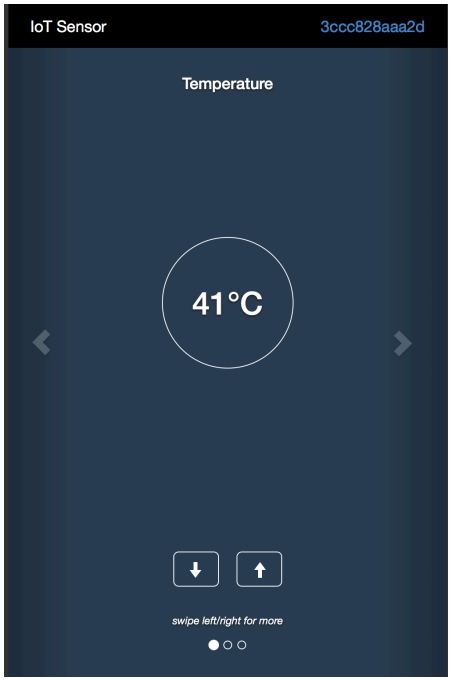
button in the top right of the screen. This will save and deploy your changes.
11. Click on the debug tab in the right-hand pane. Every second, the temperature is received in the Node-RED application, and triggers the flow. Since this flow has two green debug nodes, the debug tab displays the contents of the messages passed into the debug node. This is helpful when something doesn't work right and you can see the values being passed around.



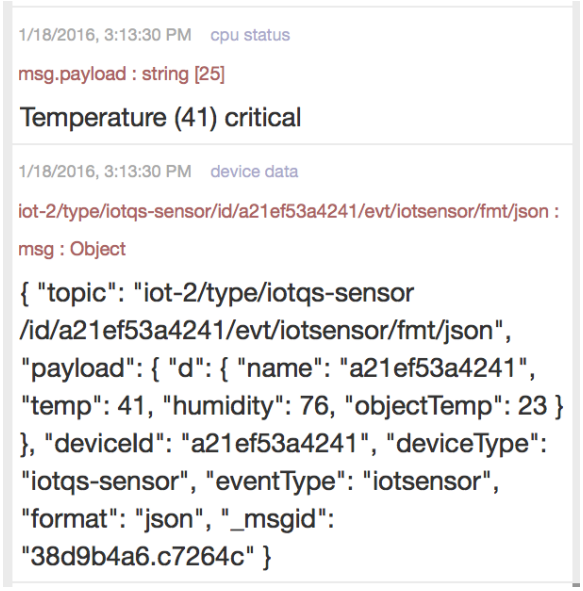
12. The yellow node is called a switch node. You can program logic using a switch node and split a flow into two or more flows based on a value. In this example, if the temperature is less than or equal to 40°C, it is considered "safe" and continues with the flow to the function named safe. If the temperature is greater than 40°C, it is considered "danger[ous]" and continues with the flow to the function named danger.



13. To trigger danger, increment the temperature using the up arrow on the simulated temperature gauge.



The debug message should change.



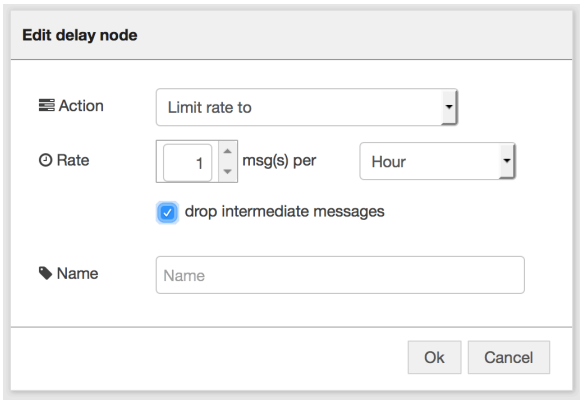
Connect to Twitter and Tweet High Temperature

This section connects to a Twitter account and uses the Twitter account to tweet when the temperature is “dangerous”. This section is optional and may be skipped.

1. Sign up for a Twitter account at `http://twitter.com`. If you already have a Twitter account, proceed to step 2.

2. Add a 

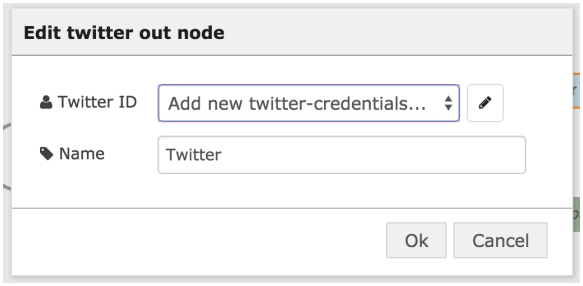
node with the following settings:

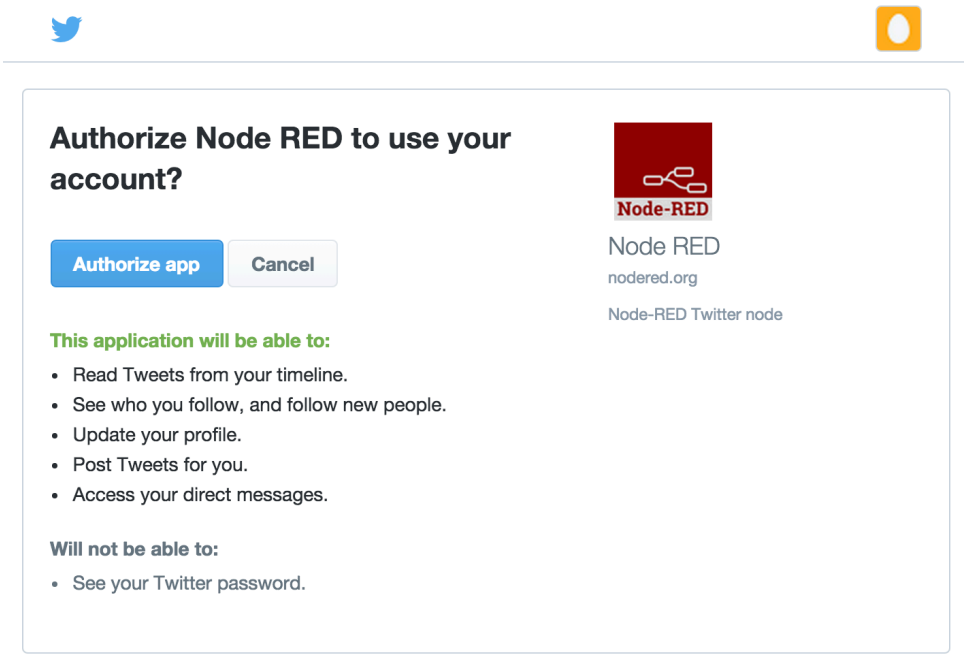


   This node limits how often the flow is run. Since the temperature is dangerous every second, without this node, a tweet would be sent every second. With this node, we can limit this flow to send a tweet once an hour.

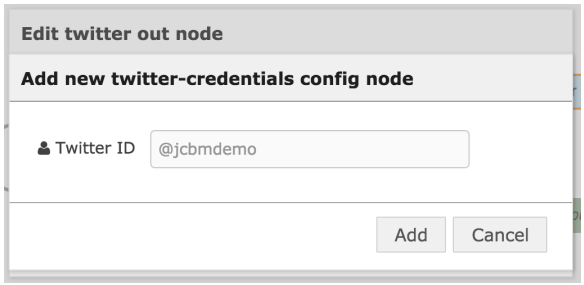
3. Add a 

node. Click on the pencil button and authenticate with Twitter. The account you sign in with will be used to send tweets.

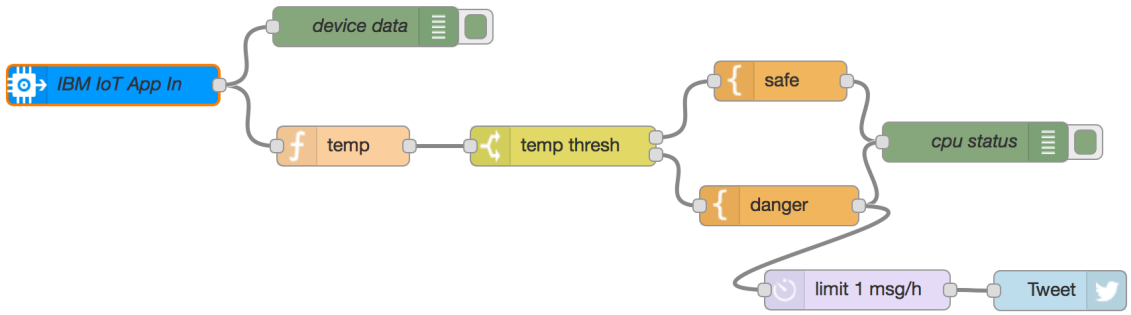




Return back to the node configuration. The settings for the Twitter node should have your username set. Click Add.



4. Connect the nodes together as follows:



5. Click on 

to save and deploy the changes.

6. Since the temperature is above 40°C, the switch statement will continue to the “danger[ous]” function, compose a message, and pass it to the Twitter node. The Twitter node uses this message as the content for the tweet.

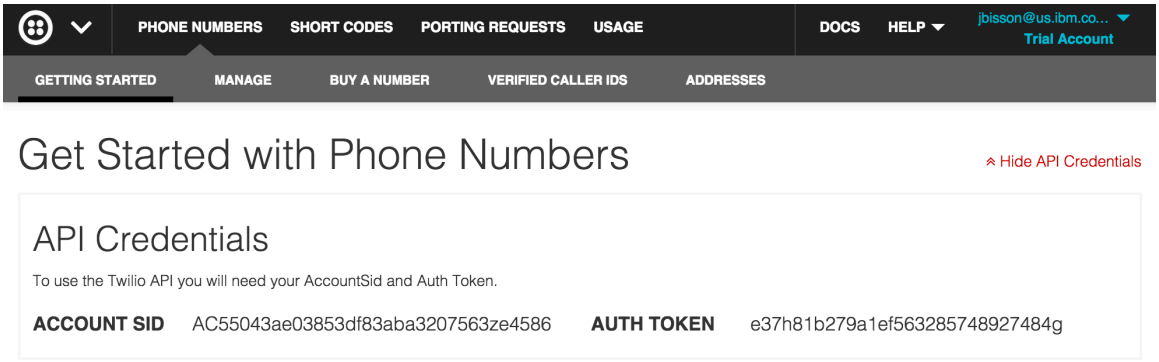
7. Visit the Twitter timeline for the user you authenticated with and verify the message has been tweeted.

Connect to Twilio and Text High Temperature

This section connects a Twilio phone number to our application and sends a text message notification when the temperature is at least 40°C. This section is optional and may be skipped.

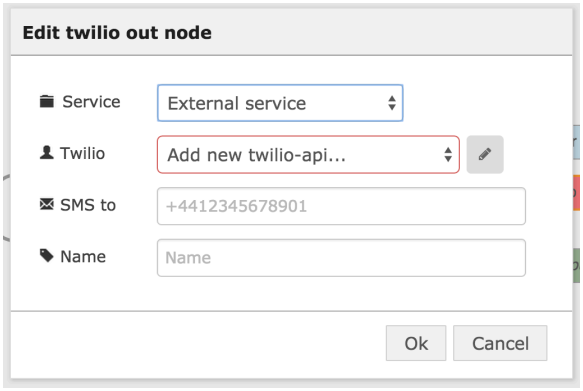
1. Sign up for a Twilio account at `http://twilio.com`. If you already have a Twilio account, sign in.

2. Go to the **Getting Started** tab, and click on **Show Credentials**.

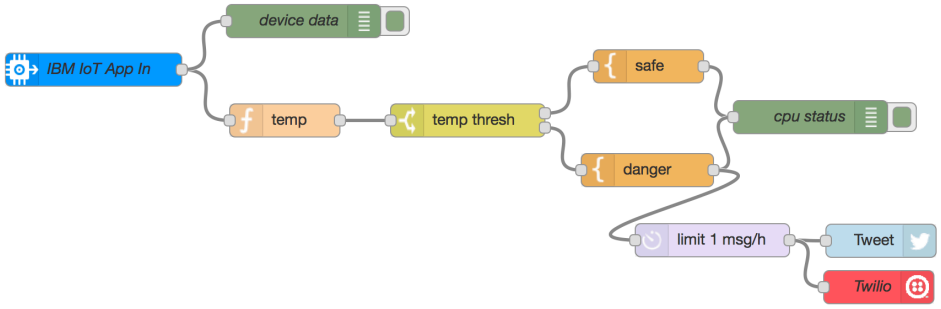


3. Add a 

node. Double click on the node to edit the configuration. Click on the Pencil to provide your **Account SID** and **Auth Token**. Fill in the **SMS to** textbox with your phone number that will be texted to.



4. Connect the nodes together as follows:



5. Click on 

to save and deploy the changes. You should receive a text message shortly.

Create a Node-RED application on Intel Edison

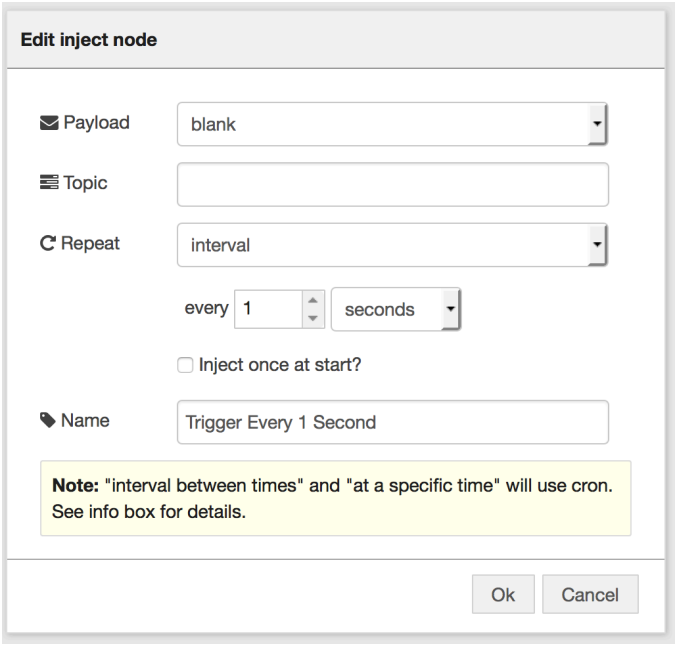
Using a simulated device is a good way to get started with the Internet of Things. When you have simulated what your device will do, then you can invest in making the hardware and hopefully reducing the cost incurred along the way. The Intel Edison board has made prototyping IoT devices easy. The Grove Starter Kit comes with a variety of sensors that can be connected to the Intel Edison and, along with custom logic, can bring developing hardware solutions down to a level anyone can build quickly and easily.

In this section, we will connect to the Node-RED application running on the Intel Edison board, capture the value of a real temperature sensor, control an LCD screen, control a LED light, and send the temperature to the IBM Internet of Things Foundation.

1. Please visit `http://ibm.biz/node-red-edison` for instructions on how to set up the Intel Edison board with the necessary Grove sensors.

2. To get started, drag a 

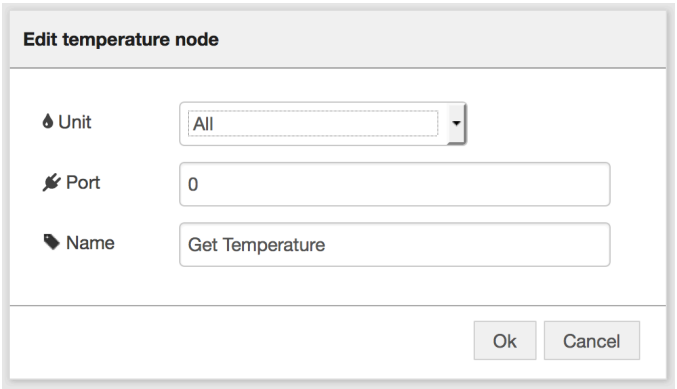
node onto the canvas. Double click on the inject node, and customize it with the following settings:



   The inject node will trigger the flow once every second.

3. Next, add a 

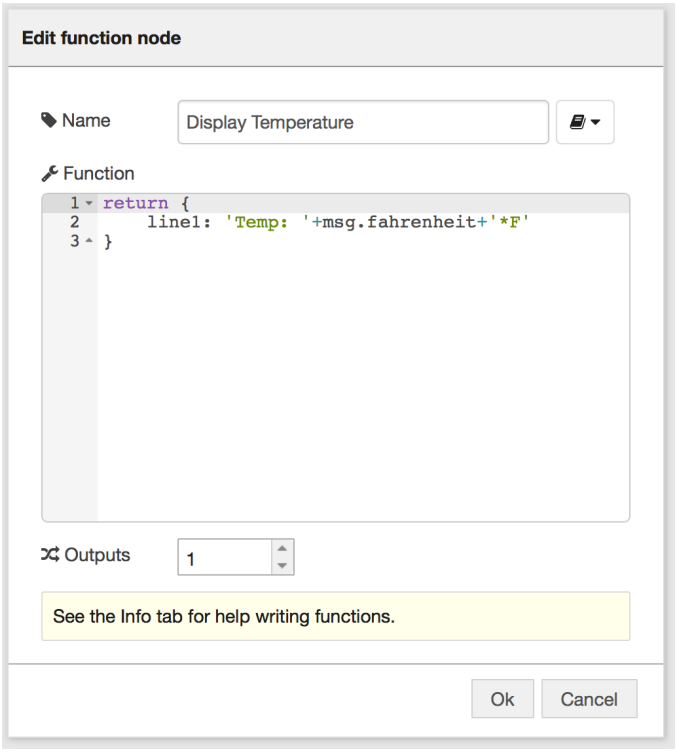
node to the right of the inject node.
4. Double click on the node and customize it with the following settings:

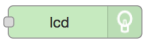


This will customize the temperature node to listen on Port A0, and return both Celsius and Fahrenheit temperature values.

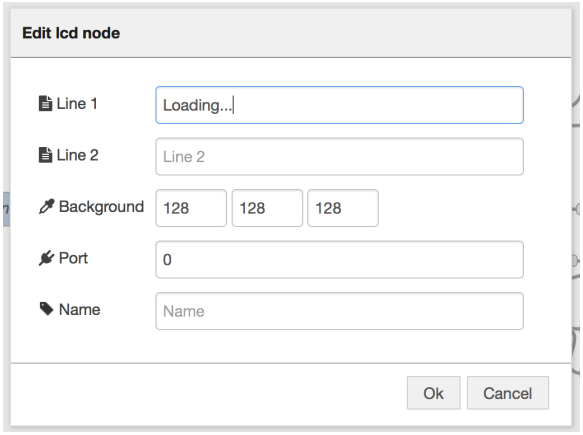
5. Add a 

node with the following settings:



6. Add a 

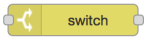
with the following settings:



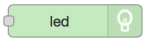
This will display the temperature value on the LCD connected to the I2C port.

7. Connect the nodes together as follows:

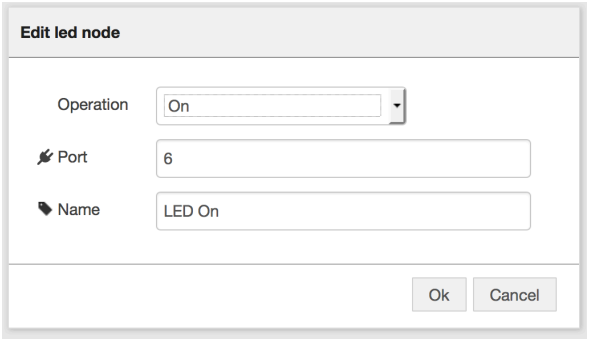
8. Click on the 

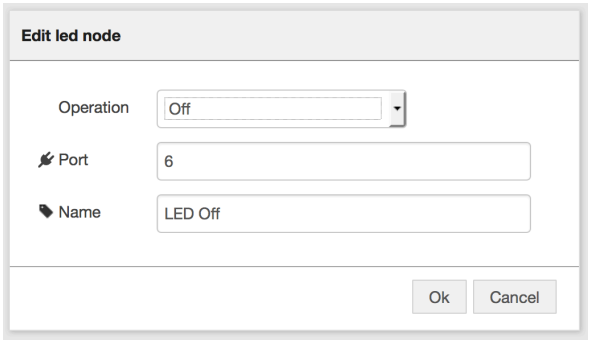
button in the top right of the screen. This will save and deploy your changes. The LCD screen should display the temperature in Fahrenheit. The temperature value screen will update once every second when the flow is activated by the inject node.
9. Next, we will activate a LED light when the temperature exceeds 80°F. Add a 

node and connect it to the temperature node. Use the following settings for the switch node:

10. Add two 

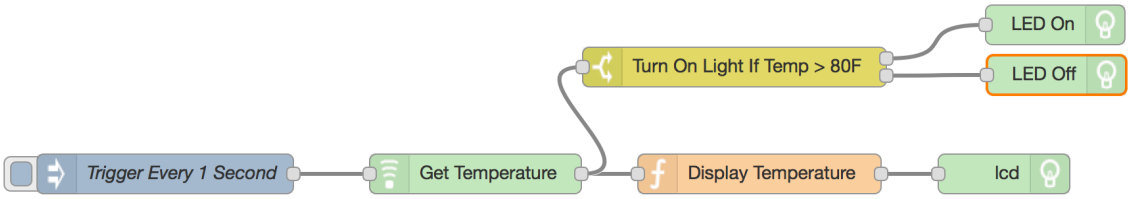
nodes, one to turn the LED on, and one to turn the LED off. Use the following settings:





This will control an LED light connected to port D6.

11. Connect the nodes as follows:



12. Click on the 

button in the top right of the screen to save and deploy your changes. When the temperature reaches above 80°F, the LED light will turn on. Otherwise, the LED light will turn off.

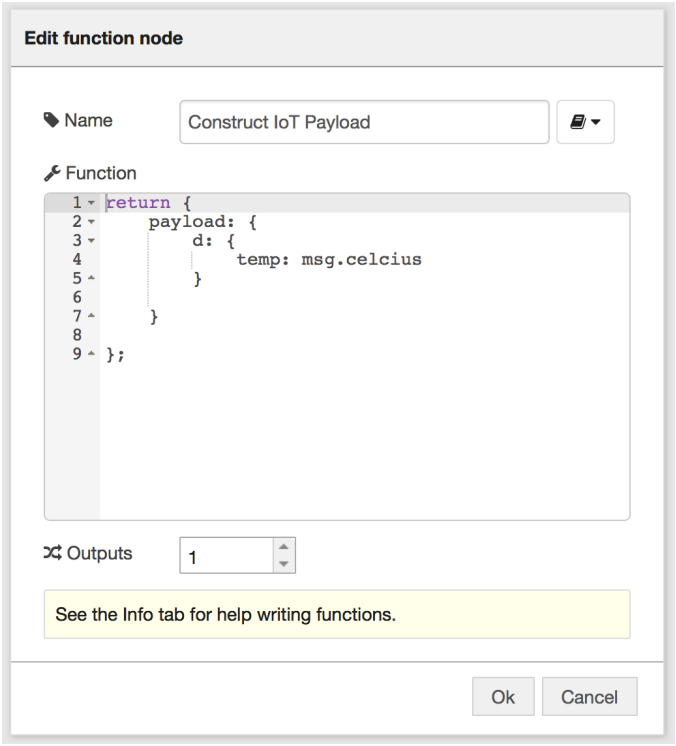
    In the next section we will connect this temperature sensor to the Cloud.

Connecting temperature sensor to IoT Foundation

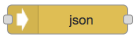
In this section, we will connect the Node-RED application reading the temperature sensor to the IBM Internet of Things Foundation service and submit the temperature value so our Node-RED application in IBM Bluemix react to high temperatures.

1. Add a 

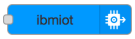
with the following settings:

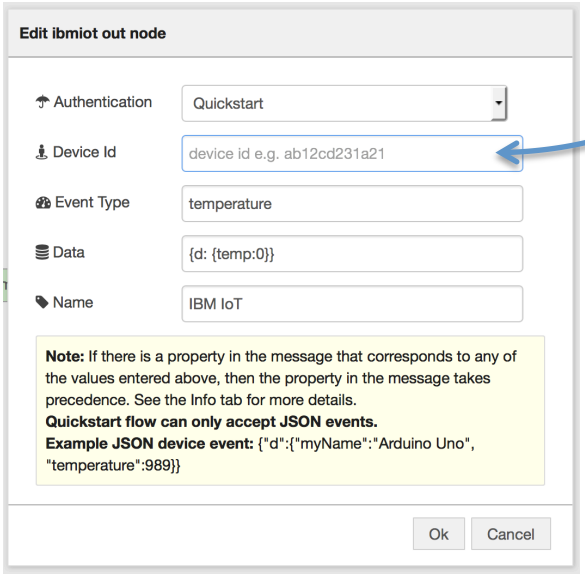


   This JavaScript function constructs a JSON object. The JavaScript object has a property named payload, which has property named d, which contains a property named temp. We use the Celsius value from the temperature sensor for the value of temp.

2. Add a 

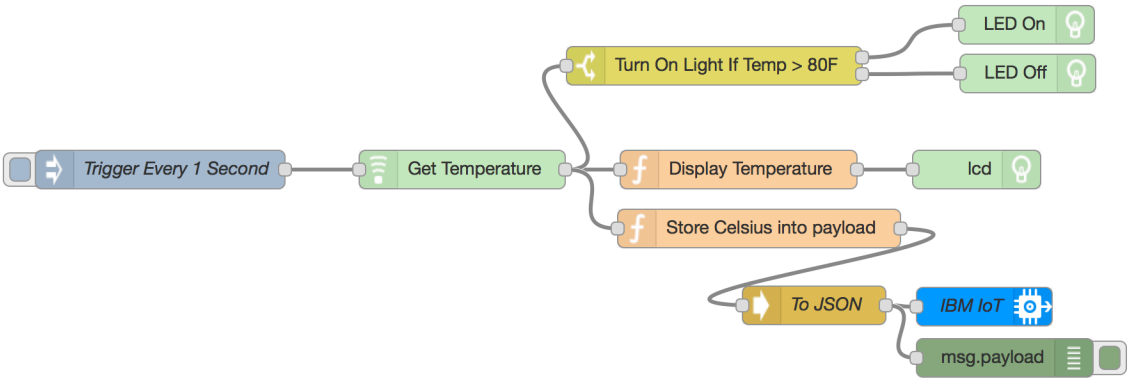
node. This node does one of two things. When passed a JSON string, it will try to parse it into a JSON object. When passed a JavaScript object, it returns a JSON string. Since we're passing in a JavaScript object, it will return a JSON string.

3. Add a  node with the following settings:



Reuse Device ID from simulated IoT sensor

4. Add a  node. Connect the nodes together as follows:



5. Click on  to save and deploy your changes.

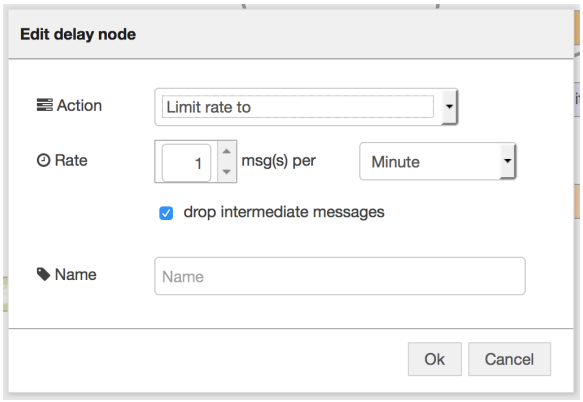
Go back to the Node-RED application in IBM Bluemix. You should see the real temperature being reported. You can close the simulated IoT temperature sensor.

Store Temperature Into Cloudant NoSQL Database

This section adds a Cloudant NoSQL database to the application and stores temperatures reported (one per minute). This functionality can be useful to run historical analysis (outside the scope of this lab) or find patterns over time. This section is optional and can be skipped. However, it is a prerequisite for the **Retrieve Temperatures From Cloudant NoSQL Database** section.

1. Add a 

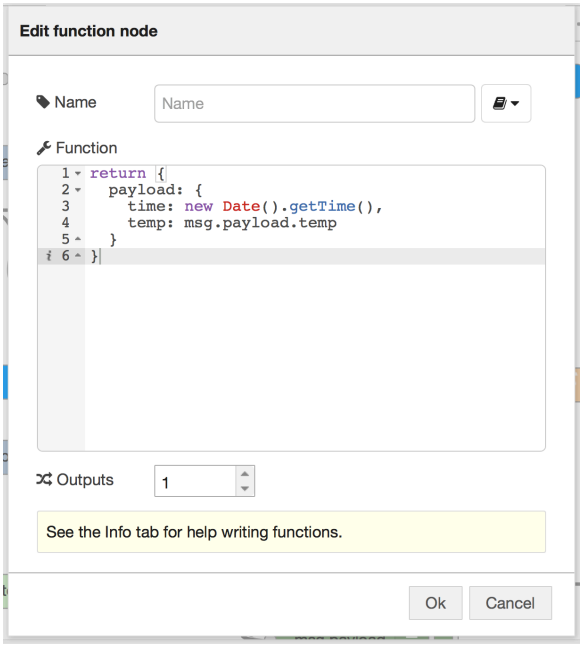
node with the following settings:

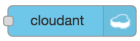


   This node will limit this flow to execute once per minute.

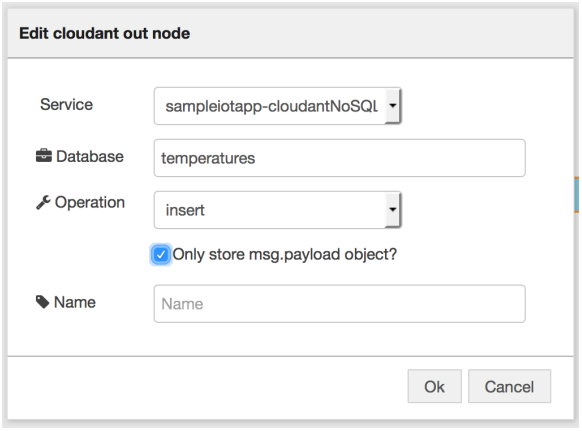
2. Add a 

node with the following settings:

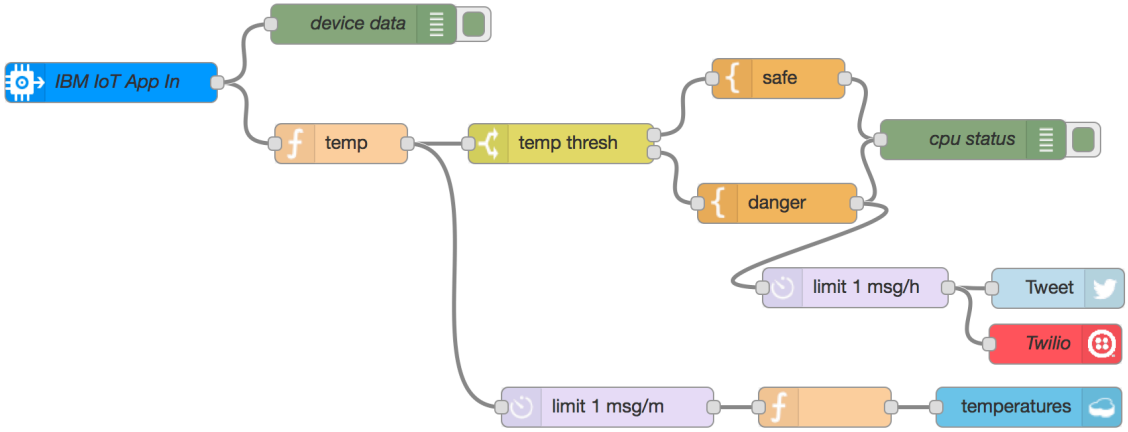


3. Add a 

node with the following settings:



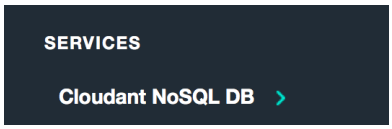
4. Connect the nodes together as follows:



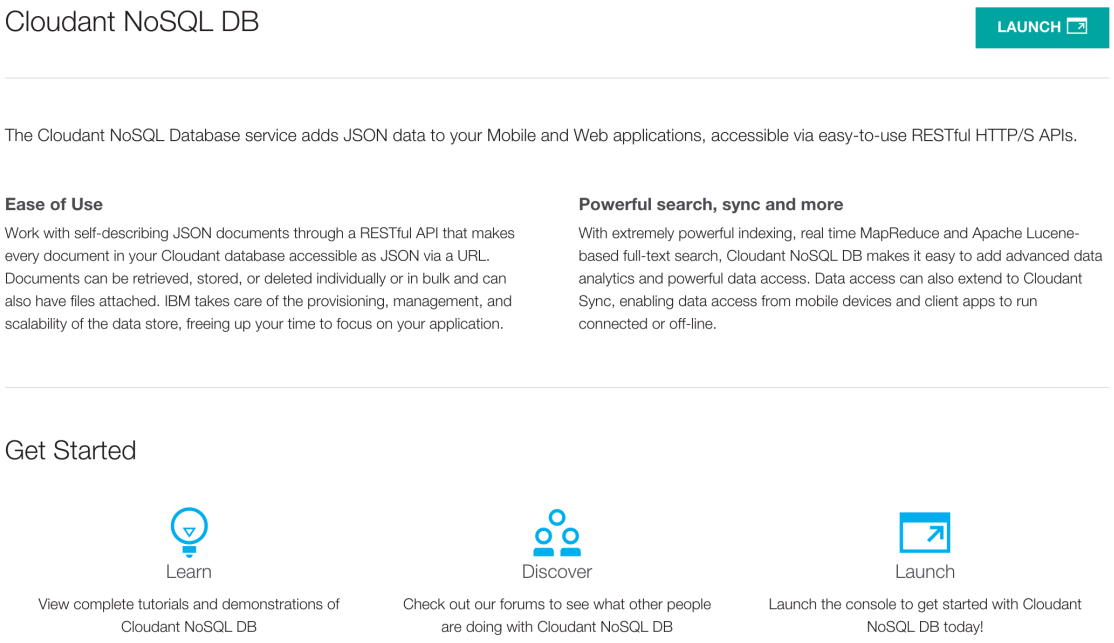
5. Click on 

to save and deploy your changes.

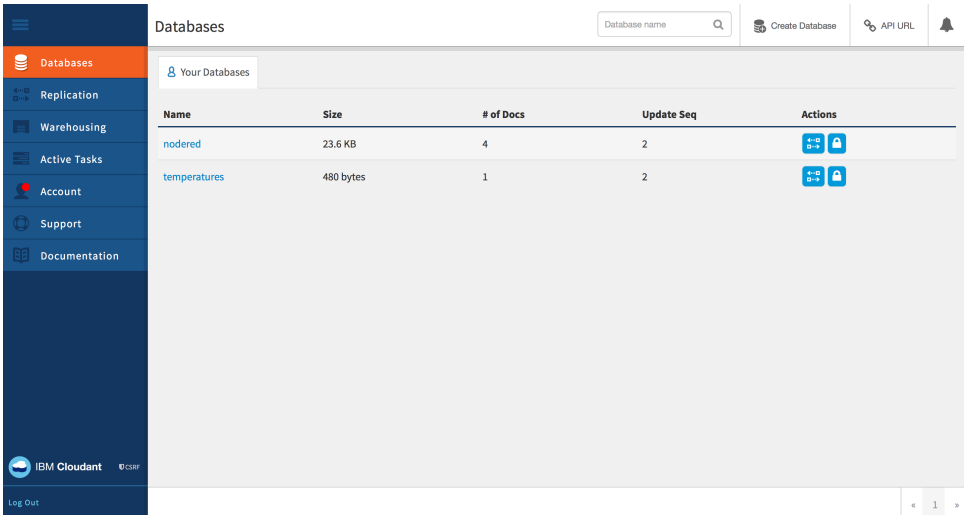
6. Go back to the IBM Bluemix dashboard and the app overview. Click on the **Cloudant NoSQL DB** link in the sidebar.



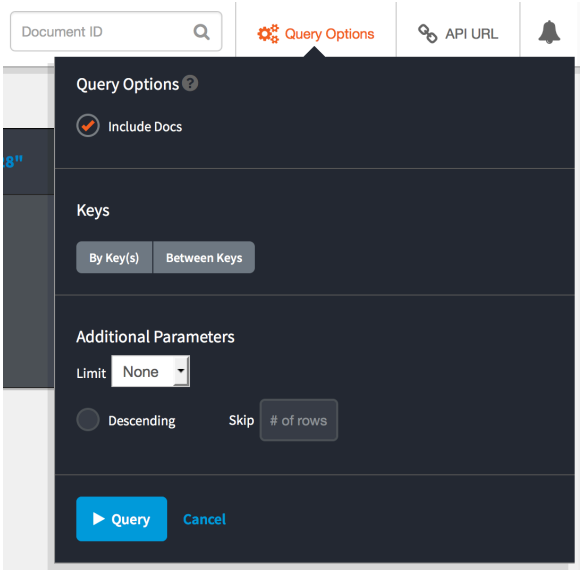
7. Click on the green **Launch** button.



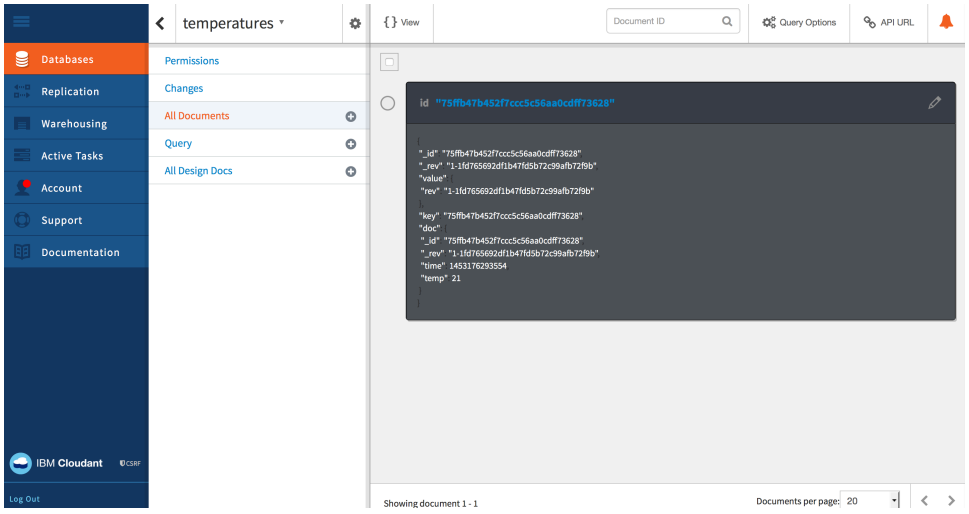
8. This is the Cloudant NoSQL database dashboard. A list of databases is displayed. The database named temperatures contains documents representing each temperature event that has been stored by the application. Click on the database named **temperatures.**



9. Click on **Query Options**, check the box next to **Include Docs**, and click on the blue **Query** button.



10. Each box represents one document (one temperature event) that contains the payload (time and temperature) we stored earlier.



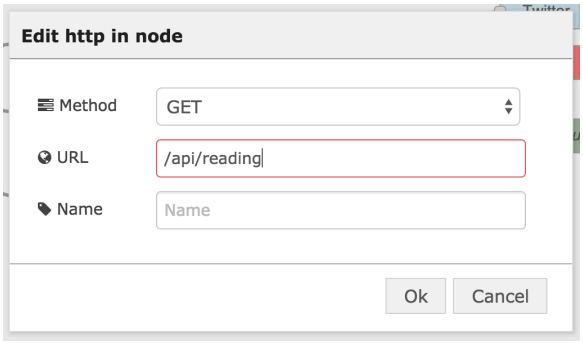
Retrieve Temperatures From Cloudant NoSQL DB

This section retrieves data from a Cloudant NoSQL database and exposes it as a HTTP endpoint. This functionality can be useful to run historical analysis (outside of the scope of this lab) or find usage patterns over time. This section is optional and can be skipped. Completion of the section titled **Store Temperature Into Cloudant NoSQL Database** is required before beginning this section.

Now that we have a Cloudant NoSQL database containing reported temperatures, let's expose the data as an HTTP endpoint.

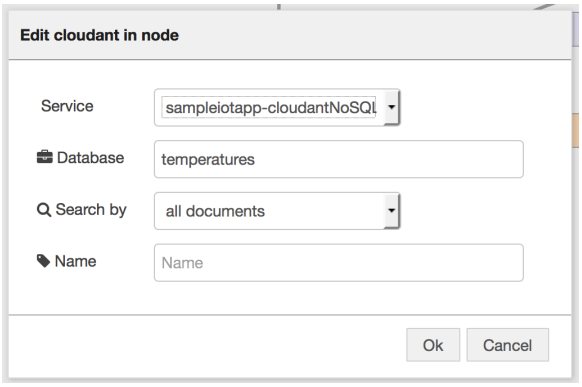
1. Add a 

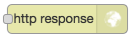
node with the following settings:



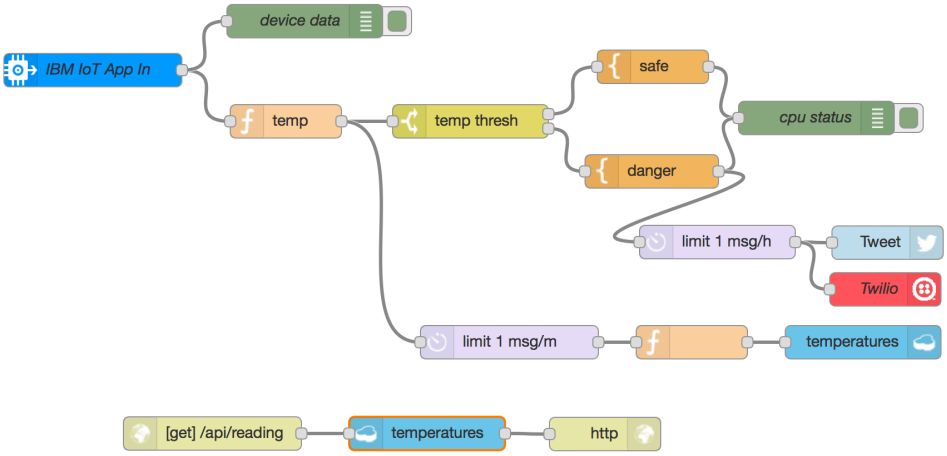
2. Add a 

node with the following settings:



3. Finally, add a 

node. Connect the nodes together as follows:



4. Click on  to save and deploy your changes.

5. Open a browser tab and visit your application's URL, appended by `/api/reading`. If you chose ***myapp*** when setting up your application, the URL would be:

   `https://`***`myapp`***`.mybluemix.net/api/reading`

   You should see data returned similar to the following:


```
[{"_id":"11d0f324f4fe069825b2170198b48a04","_rev":"1-32b465fb952bea4f5196a9e654a1514e","time":1453176413834,"temp":21},
{"_id":"11d0f324f4fe069825b2170198d178bf","_rev":"1-b61bfff2e7bbf5638618aa7671013b53","time":1453176654950,"temp":21},
{"_id":"16786b564a223659b3ad69dfebae992e","_rev":"1-a826af50277e9849845ff0eed6e58c8b","time":1453176353627,"temp":20},
{"_id":"56e2bee50c7bbe93a22fab349438ab99","_rev":"1-6fa0f11dc0aa799bb4b48fc15af62dbf","time":1453176895907,"temp":20},
{"_id":"75ffb47b452f7ccc5c56aa0cdff73628","_rev":"1-1fd765692df1b47fd5b72c99afb72f9b","time":1453176293554,"temp":21},
{"_id":"c3516c0143f46eb63a054f743bb8ef66","_rev":"1-655f173446fcaccd5d7b9992d0e9f87c","time":1453176534279,"temp":20},
{"_id":"cc20392d54dd3b131646e7db68dd78fc","_rev":"1-c10c4410806e3922c95bd7b44fbf55fd","time":1453176474098,"temp":20},
{"_id":"ef1fa6d54f3e16c0e6b763cfd105f43a","_rev":"1-0aab58b8d693dcaac314f86e0a926866","time":1453176835565,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c11fffa9","_rev":"1-961cdac2e2fd0877480670dee961c94c","time":1453176594442,"temp":21},
{"_id":"f8c256cc283e0ab5e72420d9c130f876","_rev":"1-852900329df8aace34f9daaac6135002","time":1453176715108,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c1356c14","_rev":"1-03ecf88ac6c3c38609232e16c13b85ef","time":1453176775474,"temp":20},
{"_id":"f8c256cc283e0ab5e72420d9c146897f","_rev":"1-6f69e480c133d5990d55257fbddb9fbb","time":1453176918219,"temp":21}]
```


   This data is in a format called JSON, and you can use this data in web applications. As you inject more data in the IoT example in Node-RED, this dataset will expand to include that data.

What's next?

Congratulations! You have completed this lab. Now it's your turn to make something new! Take a few minutes and think about what you could build with the components you've learned in this lab. Use experiences that you've had at school, with friends and family, and with the technology (or non-technology) around you. If you need more room to answer a question, feel free to use another sheet of paper.

1) What could you build with a temperature sensor, LED light(s), and a LCD screen? Think outside the box, there's no right or wrong answer! Describe what other materials (sensors, hardware, materials, fabrics, etc.) you might need.

2) Where could temperature sensors be placed and what else could be measured?

3) What other sensors might be connected to the Intel Edison (or other IoT device) and how do you envision them working together?

4) Describe who would use your concept. Would they pay once, or would there be a reoccurring subscription for your product?

5) When do you think Internet of Things will be commonplace in the world around us? What would your plan of action be to be part of this evolution?

Doh! I should have asked that! Ask yourself a question! Make it a good one!

Question: _____
Your answer: