

CASCON 2017

November 6th-8th

Using IoT and cognitive services to provide a personalized experience

Authors:

Cesar Ivan Orozco Cervantes, Software Engineer, Cesar.Cervantes@hcl.com

Heath Thomann, Software Engineer, Heath.Thomann@hcl.com

Table of Contents

1. Introduction.....	2
2. Before you go... ..	3
2.1 Prerequisites.....	3
2.2 Lab overview and software used	4
3. Instructions.....	4
3.1 Create a Node-RED boilerplate in BlueMix	4
3.2 Create the Twitter and Watson Tone Analyzer Node-RED application	5
3.2.1 Configure the Twitter node	5
3.2.2 Configure the Watson Tone Analyzer service	6
3.2.3 Configure the Watson Tone Analyzer node	7
3.2.4 Connect (wire together) the Twitter and Watson nodes	8
3.2.5 Process the tone analyzer output	9
3.2.6 Save the highest tone	10
3.2.7 Add debug nodes to view tweet and tone data	11
3.2.8 Let's test our work so far	12
3.2.9 Create an HTTP request/response for external application	13
3.2.10 Test the HTTP request/response	15
3.2.11 Connect to an IoT device and create the IoT Node-RED flow	15
3.2.12 Connect to an IBM IoT sensor	15
3.2.13 Create the IoT Node-RED flow	16
3.2.14 Create an HTTP request/response for external application	18
3.2.15 Test the HTTP request/response	21

3.3	Building the mobile application	21
3.3.1	Accept the RAD Beta License Agreement	21
3.3.2	Create a Bluemix server to deploy the mobile application	21
3.3.3	Clone the mobile project	22
3.3.4	Edit the Source Code	23
3.3.5	Test the application locally	24
3.3.6	Deploy the application to Bluemix.....	24
3.3.7	Test the solution	25

1. Introduction

Every day the world becomes more and more connected. Everything from the cars we drive, the phones we use, devices we wear, and to the factories producing our products. These devices can be linked together and to a huge network of data. It is estimated that by 2020 there will be 50 billion Internet of Things (IoT) devices and an estimated 90% of cars will be connected.

Taking advantage of IoT we can build applications that provide a more personalized experience for each individual user, not only by displaying personalized content, but also by changing the look and feel of the application itself to adapt it to how people “feel” about certain topics, this can be achieved by using the cognitive capabilities of Watson to analyze social networks, in combination with physical sensors we can personalize the experience even more by using information from the environment. Large companies and independent developers can implement applications that use this combination of technologies to create a new generation of applications, making both, the products offered by these applications and the applications themselves more attractive to users.

By implementing this system, users will have the possibility to enter a set of keywords, Watson can analyze social networks (user’s own posts or the social network in general) using this keywords to determine how to change the look and feel of the application, for example, if Watson detects that users are upset, using color psychology, it can automatically change the colors of the application in an attempt to influence the mood of the users, combining this system with light sensors, the application can adapt the colors chosen by Watson to the current light conditions in the user’s environment. SimpleLink SensorTag by Texas Instruments is a relatively low cost IoT-ready device designed for early adopters and IoT enthusiasts to start experimenting. It has sensors for different purposes: Infrared thermopile temperature, 9-axis motion, altimeter/pressure, ambient light, magnet and humidity sensors. It works under

CASCON 2017

Bluetooth Low Energy (BLE) technology supported by Bluetooth 4.0 and is compatible with Beacons and Bluetooth Smart. The Texas Instruments BLE mobile application allows you to quickly put the data gathered by these sensors right onto the IBM IoT Foundation (IBM IoT Foundation) cloud, either on the Quick Start boilerplate or well in a custom IoT application powered by IBM IoT Foundation.

While IoT is still a buzz word, it is matured enough that most people have a general idea what it is, and what a 'thing' is. But what might not be as clear is how to connect to an IoT 'thing' to do something useful. That is, the concept of IoT might still be somewhat nebulous, and using it to one's advantage is still a mystery for most people.

Given the proliferation of small, low consumption sensors, there is a need for software that can be used to create IoT applications that can tie together devices and its data to users. Enter Node-RED. Node-RED is used to create IoT applications. It is a Flow-based Programming tool, meaning it is a network of connected "nodes" where each node has a specific purpose. For example, a node can be created to listen to a sensor on a device, such as a SensorTag, and pass the sensor data to a temperature function node which might act upon the data at a certain threshold by sending the data to an email, or twitter, node. The network of connected (wired) node is consider a flow.

In this workshop, the attendee will learn how to connect an IoT device to the IBM IoT Foundation (running on IBM Bluemix) to retrieve its data and combining it with Watson services to personalize the experience in the application. A mobile / web application, that displays relevant information to the user, and a Bluemix application, running the backend and gathering the data from IoT Foundation, will be developed using Rational Application Developer.

2. Before you go...

2.1 Prerequisites

You will need:

- An IBM Bluemix account. If you do not have one yet, go to <https://ibm.biz/BluemixSignUp> and sign up for one.
- A Twitter account. This can be your personal account, or one you created just for this event.
- Very basic familiarity with Node-RED (although an intro/overview will be given in the lab): nodered.org
- Very basic familiarity with the Texas Instruments SensorTag: http://www.ti.com/ww/en/wireless_connectivity/sensortag/

CASCON 2017

2.2 Lab overview and software used

This lab will use Node-RED, BlueMix, and Rational Application Developer (RAD) software. Using Node-RED editor we will create and deploy two Node-RED flows. In general, the first flow will consist of the following:

1. A Twitter node that will connect to a user's Twitter account and retrieve tweets from the user account.
2. A node to extract the tweet text.
3. A Watson node which connects to the Watson Tone Analyzer. The tweet text will be sent to the analyzer to analyze the tone of the tweet. The analyzer will return a score for 5 different types of tones (Anger, Disgust, Fear, Joy, or Sadness).
4. A node to process the results from the Watson Tone Analyzer to find the highest score, and corresponding tone.
5. A node to save the highest tone.
6. An http request/response node to return the tone as a JSON string.

The second flow will basically consist of the following:

1. An IBM IoT node that will connect to either a Texas Instrument SensorTag IoT sensor, or a simulated sensor provided by BlueMix.
2. A node to extract and save light data from the SensorTag.
3. A node to save the light data.
4. An http request/response node to return the light data as a JSON string.

The consumer of the tone and light data from the Node-RED flows described above will be a mobile application. This application will use the tone and light data to change the color of the app text/background.

3. Instructions

3.1 Create a Node-RED boilerplate in BlueMix

In this section, we will create the Node-RED server on BlueMix.

1. Click the following link to go to open BlueMix:
<https://console.bluemix.net/dashboard>
2. Log into your BlueMix account.
3. Once logged in, select the **Catalog** link in the upper right-hand corner.
4. In the **Search** bar enter **Node-RED**.
5. In the results, select **Internet of Things Platform Starter**.
6. In the **Create a Cloud Foundry App**, enter an app name of **cascon-nodered**.
7. In the **Host name field**, edit the name to create a unique host name. I would suggest we all use a host name in the form cascon-nodered-<lastname>.
8. Take the defaults for the rest of the page, and click the **Create** button.

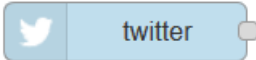

CASCON 2017

9. Be patient, after hitting create the browser might spin for a minute while BlueMix creates the app.
10. Eventually a new screen should open indicating the app is starting. Have more patience.
11. Eventually the app will start and a “**Visit App URL**” will be available.
12. When available, click “**Visit App URL**”.
13. In the welcome page, click **Next**.
14. In the **Secure your Node-RED editor**, select the **Not Recommended** option. For demo purposes, we don’t care if our editor is unsecure.
15. Tick the box to confirm you want your editor to be insecure.
16. Click **Next**.
17. Click **Next** in the next page.
18. Click **Finish** in the next page.
19. After the settings have been applied, click **Go to your Node-RED editor**.
20. Conveniently, BlueMix creates an initial flow for you. As nice as that is, we don’t want it. Select all the nodes and click the delete button. Then click the **Deploy** button.
21. Take note of the URL of your Node-RED editor. In other words, the URL will look something like:
https://cascon-nodered-<lastname>.....mybluemix.net/red/#flow/<flow number>
22. In the steps below, we will make use of the portion: https://cascon-nodered-<lastname>.....mybluemix.net

3.2 Create the Twitter and Watson Tone Analyzer Node-RED application





We can now go forward with developing the Node-RED flow.

3.2.1 Configure the Twitter node

1. Under the **social** section within the palette (the palette is on the left-hand side of the Node-RED editor), find the twitter node: 
2. Drag-and-drop the twitter node into the flow workspace.
3. Double-click the twitter node.
4. Click the pencil icon: 
5. Click the button labeled **Click here to authenticate with Twitter**.
6. In the resultant page (pop up browser), log in to Twitter (if you are not already logged in) and click the **Authorize app** button.

CASCON 2017

7. Close the pop up browser page to return to the browser containing your Node-RED editor. Your Twitter handle should be filled in next to the **Twitter id** text field.
8. Click the **Add** button.
9. In the **Search** pull down, select **the tweets of specific users**.
10. In the **User** text field enter your twitter handle
11. In the **Name** text field, enter a name of your choice (e.g. My tweets). The configured node should look as follows (my twitter handle is @heath_hcl):

 Twitter ID	<input type="text" value="@heath_hcl"/>
 Search	<input type="text" value="the tweets of specific users"/>
 User	<input type="text" value="@heath_hcl"/>
 Name	<input type="text" value="My tweets"/>

12. Click the **Done** button.
13. Click the **Deploy** button to deploy the flow to the server.
14. Leave your browser containing Node-RED editor open, after the next steps below we will come back to it.

3.2.2 Configure the Watson Tone Analyzer service

1. Click the following link to go to BlueMix:
<https://console.bluemix.net/dashboard/watson>
2. Click the **Create Watson service** button.
3. In the search bar enter **tone analyzer** and click **filter**.
4. In the search results, click the **Tone Analyzer**.
5. The **Lite** plan should be selected by default, if not, select it and click the **Create** button.
6. In the resultant page, click **Service Credentials** on the left-hand side menu.

CASCON 2017

7. Under the **Actions** column click **View credentials**. This gives the URL to use to connect to the analyzer, as well as the username and password.
8. Take note of the credentials. They should look something like this (the username and password will be different of course):

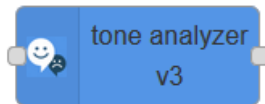
```
{
  "url": "https://gateway.watsonplatform.net/tone-analyzer/api",
  "username": "cb38dd76-9690-4e38-a723-0f9fd950106f",
  "password": "mruMaC6rPrA4"
}
```

9. Leave open the browser to the credentials page, as we will come back to it in the steps below.

3.2.3 Configure the Watson Tone Analyzer node

Now that we have created the tone analyzer service, we can continue with the Node-RED flow and configure the tone analyzer node. Return to the browser containing your Node-RED editor.

1. In the Node-RED editor, look under the **IBM Watson** section in the palette and












find the tone analyzer node:

2. Drag-and-drop the tone analyzer node into the flow workspace (drop it next to the recently configured twitter node).
3. Double-click the analyzer node.
4. In the **Name** field, enter a name, for example, enter My tone analyzer.
5. For the **Username** field, enter the username provided by the Watson Tone Analyzer service from the steps above (e.g. cb38dd76-9690-4e38-a723-0f9fd950106f from above).
6. For the **Password** field, enter the password provided by the Watson Tone Analyzer service from the steps above.
7. Ensure that the **Use Default Service Endpoint** check box is checked.
8. In the **Method** box, select **General Tone**.

CASCON 2017

9. In the **version_date** enter **Multiple Tones**.
10. In the **Tones** box, select **Emotion**.
11. In the **Sentences** box, select **true**.
12. In the **Content type** box, select **Text**.
13. The configured node should look as follows:

 Name	<input type="text" value="Analyze tweet tone"/>
 Username	<input type="text" value="cb38dd76-9690-4e38-a723-0f9fd950106f"/>
 Password	<input type="password" value="••••••••"/>
<input checked="" type="checkbox"/> Use Default Service Endpoint	
 Method:	<input type="text" value="General Tone"/>
 version_date:	<input type="text" value="Multiple Tones"/>
 Tones	<input type="text" value="Emotion"/>
 Sentences	<input type="text" value="True"/>
 Content type	<input type="text" value="Text"/>
 Input Text Language	<input type="text" value="English"/>

14. Hit the **Done** button.
15. Click the **Deploy** button to deploy the flow to the server.

3.2.4 Connect (wire together) the Twitter and Watson nodes

Now that the two nodes are fully configured, we need to connect them.

1. From the twitter node output (the small grey circle on the right-hand side of the node), left mouse click and hold.

CASCON 2017

2. Drag a line from the twitter output to the tone analyzer input. The entire flow to this point should look as follows:



3. Click the **Deploy** button to deploy the flow to the server.

3.2.5 Process the tone analyzer output

The tone analyzer will return a score for 5 different types of tones (Anger, Disgust, Fear, Joy, or Sadness). We want to find the highest score, and corresponding tone.



1. Find the **function** node in the palette:
2. Drag-and-drop the function node into the flow workspace (drop it next to the recently configured tone analyzer node).
3. Double-click the function node.
4. In the **Name** text field, enter **Get highest tone**.
5. We need to add some code to extract the scores and tones. Copy/paste the following code into the **Function** text box:

```
var ar = msg.response.document_tone.tone_categories[0].tones;
var score = ar[0].score;
var tone = ar[0].tone_name;

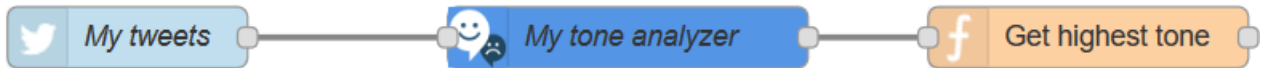
for (i = 1; i < ar.length; i++) {
  if (score < ar[i].score){
    score = ar[i].score;
    tone = ar[i].tone_name;
  }
}

msg.payload="The highest tone is: " + tone;
msg.tone=tone;
return msg;
```

6. Click the **Done** button.

CASCON 2017

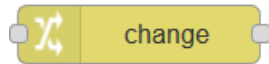
7. Connect the output of the tone analyzer to the input of the function node.
8. Click the **Deploy** button to deploy the flow to the server.
9. At this point in time, the flow should look as follows:



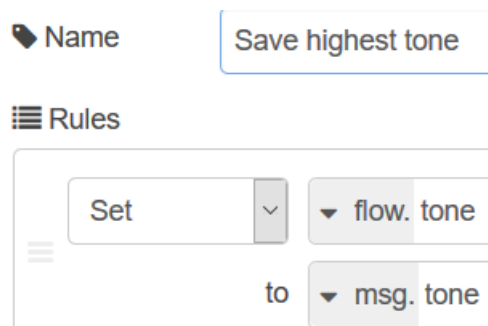
3.2.6 Save the highest tone

After finding the highest tone, the value needs to be saved for future use.

1. Find the **change** node under the **function** section in the palette:



2. Drag-and-drop the change node into the flow workspace (drop it next to the recently configured function node).
3. Double-click the change node.
4. In the **Name** text field enter **Save highest tone**.
5. Make sure **Set** is selected in the **Rules** section.
6. Define the rule as follows:

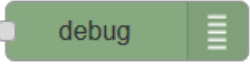


7. Click the **Done** button.
8. Connect the output of the function node to the input of the change node.
9. Click the **Deploy** button to deploy the flow to the server.
10. At this point in time, the flow should look as follows:



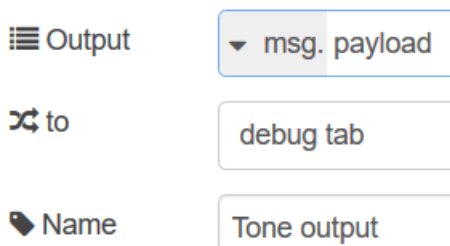
3.2.7 Add debug nodes to view tweet and tone data

If everything is configured correctly, the tweet node should be receiving the texts you send, and the analyzer should process them. It would be nice to see some output from this process. Let's add a debug node to see the tone data.

1. Find the **debug** node in the palette: 
2. Drag-and-drop the debug node into the flow workspace (drop it next to the twitter node).
3. Double-click the debug node.
4. Add **tweet.text** to the **Output** text box, and add a name of **Tweet text output** to the **Name** text field, as follows:



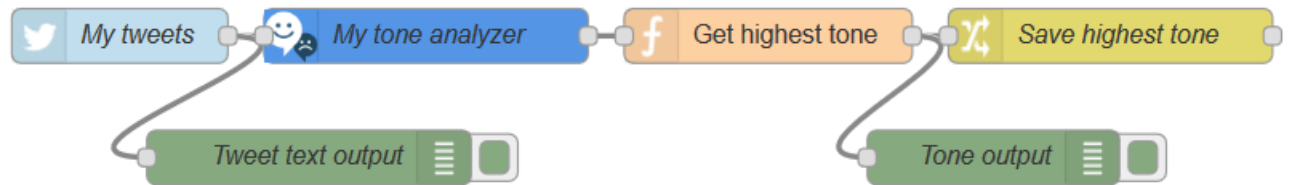
5. Click the **Done** button.
6. Connect the output of the twitter node to the input of the debug node.
7. Drag-and-drop another debug node into the flow workspace (drop it next to the function node).
8. Double-click the debug node, and this time simply add a **Name** as **Tone output**, as follows:



9. Click the **Done** button.
10. Connect the output of the function node to the input of the second debug node.

CASCON 2017

11. At this point in time, the flow should look as follows:

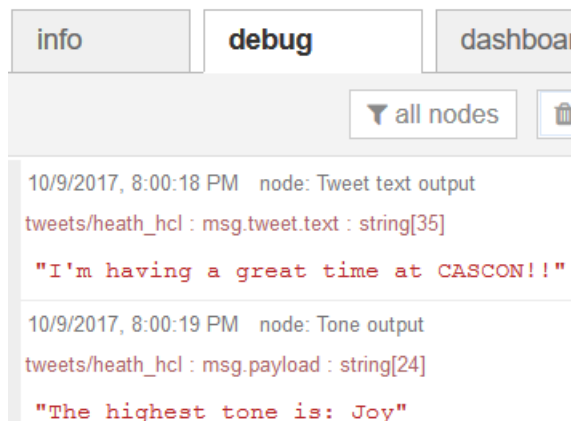


12. Click the **Deploy** button to deploy the flow to the server.

3.2.8 Let's test our work so far

Let's now have some fun and test our flow.

1. In the Node-RED editor find the **Debug** tab on the right-hand side of the editor and click it.
2. Click the following URL to open a new browser and go to your twitter account: <http://twitter.com>
3. In your twitter account enter the following tweet: **I'm having a great time at CASCON!!**
4. Go back to the Node-RED editor, and watch the Debug tab in the Node-RED editor. In about 30-60 seconds your tweet should arrive.
5. Once the tweet arrives and is processed, the debug output will look as follows:

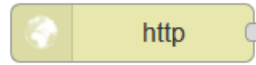


6. Let's try another one. In Twitter enter the following tweet: **I'm going to be very sad when this lab is over.**
7. After 30-60 seconds, the debug output should contain your tweet, along with the tone **Sadness**.

3.2.9 Create an HTTP request/response for external application

At this point we have saved the tone. To allow external applications access to the tone, we need to create an http request/response to allow external access. Later in the demo, our mobile application will access the tone via the http request.

1. Find the **http** node in the palette under the input section (don't use the http request node, there is a distinction):



2. Drag-and-drop the http node into the flow workspace (drop it under the existing flow).
3. Double-click the http node.
4. Configure the node as follows:


Method	GET
URL	/tone-data
Name	Get tone data


5. Click the **Done** button.
6. Find the **template** node in the palette under the **function** section:





7. Drag-and-drop the template node into the flow workspace next to the http node.
8. Double-click the template node.
9. In the **Name** text box enter a name, e.g. **JSON tone data**.
10. In the **Template** body enter `{"tone": "{{ flow.tone }}"}`
11. In **Output as** pull down, select **Parsed JSON**.
12. The configured node should look as follows:

CASCON 2017

 Name

 Set property

 Format

 Template Synt

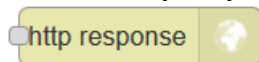
```
1 {"tone": "{{ flow.tone }}"}
```

→ Output as

13. Click the **Done** button.

14. Connect the output of the http node to the input of the template node.

15. Find the **http response** node in the palette under the **output** section:



16. Drag-and-drop the http response node into the flow workspace next to the template node.

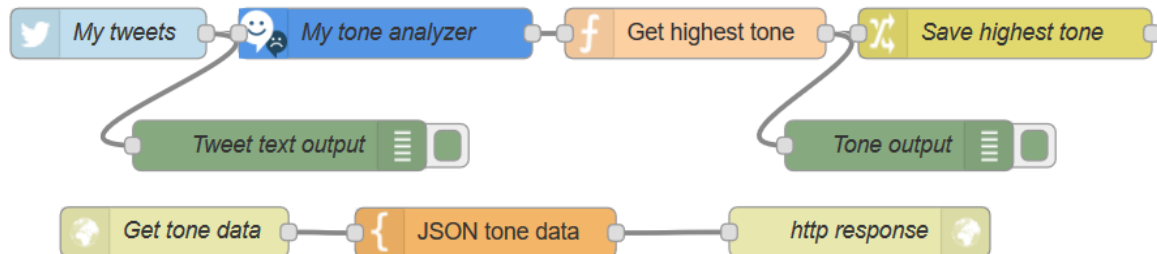
17. Double-click the http response node.

18. Simply add **http response** in the **Name** field.

19. Click the **Done** button.

CASCON 2017

20. Connect the output of the template node to the input of the http response node.
21. Click the **Deploy** button to deploy the flow to the server.
22. The entire flow should look as follows:



3.2.10 Test the HTTP request/response

1. Open a new browser and enter the URL to your BlueMix Node-RED server, less the `/red.....`, and append `"tone-data"`. That is, use:

`https://cascon-nodered-<lastname>.....mybluemix.net/tone-data`

2. The most current tone data should be returned in JSON format.

3.2.11 Connect to an IoT device and create the IoT Node-RED flow

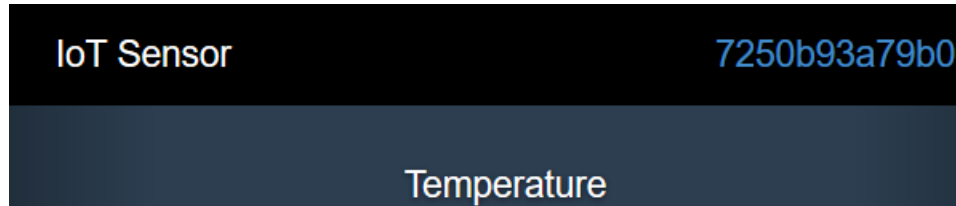
The last Node-RED flow we will create is one that will receive data from either an IoT sensor simulator, or a Texas Instruments SensorTag IoT device. There will be a few SensorTag devices provided by the instructors which will be used to collect light levels (lux) in the room. However, some participants might choose to simulate an IoT device. IBM offers an IoT device simulator that works the exact same way as the SensorTag. The following steps will use the IoT simulator. However, as will be seen below, it is simple to switch from the IoT simulator to the SensorTag devices in the lab.

3.2.12 Connect to an IBM IoT sensor

1. Click on the following link which will take you to an IBM IoT sensor simulator:
<https://quickstart.internetofthings.ibmcloud.com/iotsensor/>

CASCON 2017

2. At the top of the sensor you should see the following:



3. The number (i.e. 7250b93a79b0) in the upper right-hand corner is the sensor id. Copy this number as it will be used below when creating the Node-Red flow.
 - Note that each time you refresh the sensor web page, a new number will be given. Furthermore, this number will time-out eventually. Keep this in mind when using the sensor id.
 - As was mentioned above, a SensorTag device will be available on the day of the lab. The SensorTag will have a sensor id just like the above simulated sensor. This makes it easy to use one or the other. This will become more obvious in the steps below.

3.2.13 Create the IoT Node-RED flow

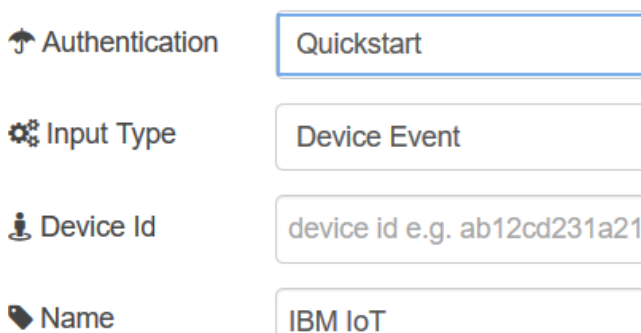
To connect to the IoT sensor from Node-RED, we will use the IBM IoT QuickStart nodes.

1. In the Node-RED editor, click the  button to create a new flow.

2. Find the **ibmiot** node in the palette under the **input** section:



3. Drag-and-drop the ibmiot node into the flow workspace.
4. Double-click the ibmiot node.
5. The ibmiot configurations will look as follows:

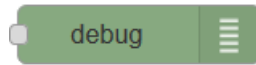


CASCON 2017

6. In the Device Id text field, enter the id from the IoT device (e.g. from the above steps the id was 7250b93a79b0). This can be the device id from the IoT sensor simulator, or from the SensorTag provided the day of the lab.

7. Click the **Done** button.

8. Find the **Debug** node under the **output** section in the palette:



9. Drag-and-drop the debug node into the flow workspace.

10. Double-click the debug node.

11. In the **Output** text field, enter **payload.d.temp** if you are using the IoT sensor simulator. If you are using the SensorTag on the day of the lab, enter **payload.d.light**. Ensure that the **msg.** is selected.

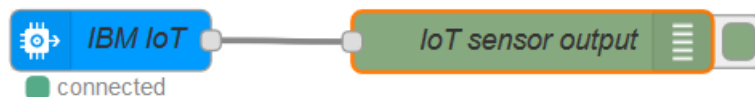
12. In the **Name** field enter **IoT sensor output**.

13. Click the **Done** button.

14. Connect the output of the ibmiot node to the input of the debug node.

15. Click the **Deploy** button to deploy the flow to the server.

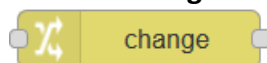
16. The flow should look as follows:



17. Find the **Debug** tab on the right-hand side of the editor and click it.

18. The data from the IoT sensor should be printed in the debug section about every second. If you connected to the IoT sensor simulator, the value set should be printed. You can go back to your browser where you connected to the sensor and change the temperature. The change should be reflected in the Node-RED debug output. If you are connected to the SensorTag, the instructor will manipulate the sensor to show how the data changes.

19. Find the **change** node under the **function** section in the palette:

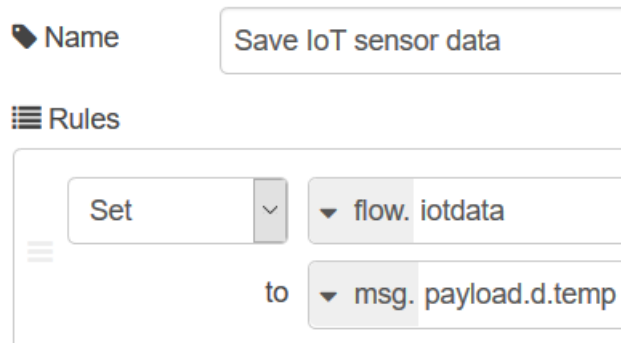


20. Drag-and-drop the change node into the flow workspace, dropping it under and near to the ibmiot node.

CASCON 2017

21. Double-click the change node.

22. In the **Name** text field enter **Save IoT sensor data**. Make the **Rules** look as follows:



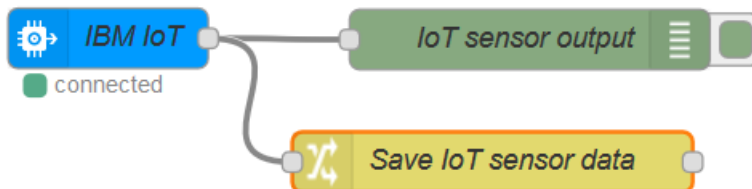
Note: if you are using the SensorTag data, use **payload.d.light** rather than **payload.d.temp**.

23. Click the **Done** button.

24. Connect the output of the **ibmiot** node to the input of the change node.

25. Click the **Deploy** button to deploy the flow to the server.

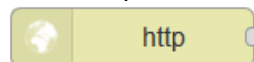
26. The flow should look as follows:



3.2.14 Create an HTTP request/response for external application

At this point we have saved the sensor data. To allow external applications access to the data, we need to create an http request/response to allow external access. Later in the demo, our mobile application will access the data via the http request.

1. Find the **http** node in the palette under the input section (don't use the http request node, there is a distinction):



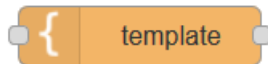
2. Drag-and-drop the http node into the flow workspace (drop it under the existing flow).

CASCON 2017

3. Double-click the http node.
4. Configure the node as follows:


Method	GET
URL	/sensor-data
Name	Get sensor data


5. Click the **Done** button.
6. **Find** the **template** node in the palette under the **function** section:





7. Drag-and-drop the template node into the flow workspace next to the http node.
8. Double-click the template node.
9. In the **Name** field enter the name **JSON sensor data**
10. In the **Template** body enter `{"lux": "{{ flow.iotdata }}"}`
11. In the **Output as** drop down enter **Parsed JSON**.
12. The configured node should look as follows:

CASCON 2017


 Name

 Set property

 Format

 Template Synt

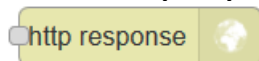
```
1 {"lux": "{{ flow.iotdata }}"}
```

 Output as

13. Click the **Done** button.

14. Connect the output of the http node to the input of the template node.

15. Find the **http response** node in the palette under the **output** section:



16. Drag-and-drop the http response node into the flow workspace next to the template node.

17. Double-click the http response node.

18. Simply add **http response** in the **Name** field.

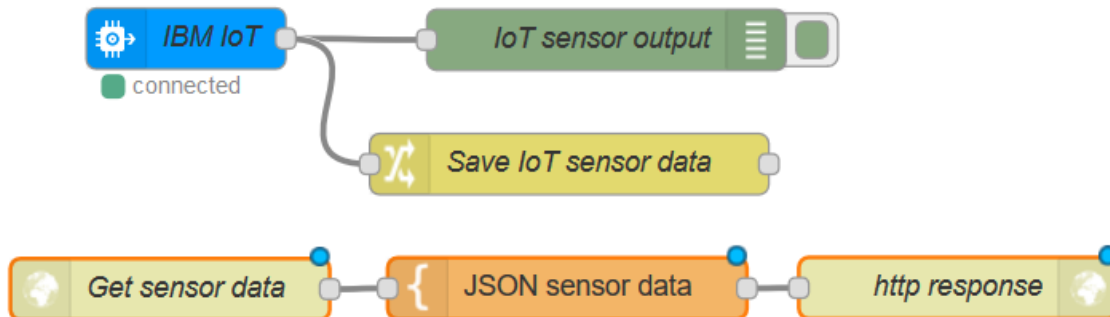
19. Click the **Done** button.

20. Connect the output of the template node to the input of the http response node.

CASCON 2017

21. Click the **Deploy** button to deploy the flow to the server.

22. The entire flow should look as follows:



3.2.15 Test the HTTP request/response

1. Open a new browser and enter the URL to your BlueMix Node-RED server, less the `/red.....`, and append `sensor-data`. That is, use:

`https://cascon-nodered-<lastname>.....mybluemix.net/sensor-data`

2. The most current sensor data should be returned in JSON format.

3.3 Building the mobile application

In this section, you will build a mobile application from a Git repository, using Rational Application Developer (RAD) and deploy it, this application will receive data from Node-RED and adjust the colors and brightness of the application based on that.

3.3.1 Accept the RAD Beta License Agreement

RAD should already be started and presented when logged into the lab machine. Bring RAD to the forefront. When RAD starts, you will be presented with a license agreement. If you agree with it enter you email and select OK. If you do not agree, you will not be able to proceed with the lab. You might also be presented with a window regarding a trial license. If you are presented with a trial license window, select ignore.

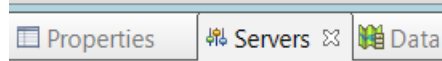
3.3.2 Create a Bluemix server to deploy the mobile application

NOTE: For those users in the lab with a full IBM BlueMix account, you can create a Bluemix server in RAD. If you have only a 30 day trial version you will not be able to

CASCON 2017

deploy to BlueMix unless you pay for it. If you are unwilling to pay for it, please skip to the next section to clone the mobile project.

1. In the RAD view section (lower portion of RAD) find the Servers tab and select it:



2. In the Servers section, right click the **Servers** view > **New** > **Server**.
3. From the list of servers, select **IBM Bluemix**, click **Next**.
4. In the **Email** and **Password** field, enter your Bluemix credentials, click **Next**.
5. Select the **dev** space and click **Next**.
6. Keep the defaults on this page and click **Finish**.

3.3.3 Clone the mobile project

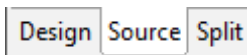
1. Go to **File** > **Import...**
2. Expand **Git**.
3. Select **Projects from Git**.
4. Click **Next**.
5. Select **Clone URI**, click **Next**.
6. Enter the URI: <https://github.com/cesarcer/cascon2017>, click **Next**.
7. In the **Branch Selection** page, select **master**, click **Next**.
8. In **Local Destination** page keep the defaults, click **Next**.
9. In **Select a wizard to use for importing projects** page select **Import existing Eclipse projects**.
10. Click **Next**.
11. In **Import Projects** page, select **CognitiveApp**.
12. Click **Finish**.
13. If presented with an option to migrate the project, please accept the migration to migrate the project by selecting **Next** then **Finish**.

CASCON 2017

3.3.4 Edit the Source Code

The application consists of 2 web pages, the main page and a configuration page, as well as a JavaScript file to dynamically change the color and brightness of the application, this application was developed using the **jQuery Mobile** framework. Let's edit the source code:

1. In the **Enterprise Explorer** view, expand the **CognitiveApp** project (the view can be found in the upper left section of RAD, or can be opened via: **Window > Show View > Enterprise Explorer**).
2. Expand the **WebContent** folder to find the **index.html** file, this is the entry point of the application.
3. Double click the **index.html** file, this will open the web page in the **Rich Page Editor**.
4. At the bottom left of the editor select **Source**, this will switch to source mode to have a better view of the application code:



5. In line **89**, replace the string **<REPLACE_HOSTNAME_HERE>** in 'toneUrl' with the location to your Node-RED URL. That is, recall that your URL should look something like:

`https://cascon-nodered-<lastname>.....mybluemix.net/tone-data`

Therefore, replace this:

```
89                                     var toneUrl = "https://<REPLACE_HOSTNAME_HERE>/tone-data";
```

with your Node-RED URL.

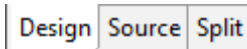
6. Repeat the previous step, but now in line **111**.
7. Now double click the **settings.html** file to open it in the editor and switch to the **Source** tab.
8. Repeat the steps (4-5) to replace **<REPLACE_HOSTNAME_HERE>**, but now in line **76**.

CASCON 2017

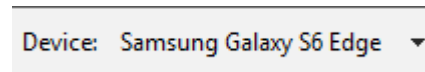
9. Repeat the steps (4-5) to replace **<REPLACE_HOSTNAME_HERE>**, but now in line **93**.

Now let's switch to Design mode to have a quick view of how the application looks:

10. Open the **index.html** file.
11. At the bottom left of the editor select **Design**, this will switch to design mode to have a better view of the web page.



12. In the top right corner of the editor select a **Device**, this will render the preview as if it was running on that device.



13. Now double click the **settings.html** file to open it in the editor, and repeat the previous two steps

3.3.5 Test the application locally

Now that we have the project in the workspace we can first test it locally to make sure it is working as expected, testing locally helps to reduce deployment times. We will deploy the application to the **Web Preview Server** (this server is part of Rational Application Developer and you can use it to quickly deploy and test your applications):

1. Right click the **index.html** file > **Run As** > **Run on Server**.
2. Expand the **127.0.0.1** folder and select **Web Preview Server**, click **Next**.
3. Keep the default on this page and click **Finish**.

This will launch the application in the **Internal Web Browser**. Using the local **Web Preview Server**, you can easily make changes in the source code and just refresh the browser to quickly test it without waiting for the application to be deployed to a remote server.

3.3.6 Deploy the application to Bluemix

NOTE: For those users in the lab with a full IBM BlueMix account, you can create a Bluemix server in RAD and deploy to it. If you have only a 30 day trial version you will not be able to deploy to BlueMix unless you pay for it. If you are unwilling to pay for it, you can only be able to run the application locally.

CASCON 2017

Now that we know the application works as expected we can deploy to the cloud.

1. Right click the **index.html** file > **Run As** > **Run on Server**.
2. Expand the **Cloud** folder and select **IBM Bluemix**, click **Next**.
3. Keep the default on this page and click **Finish**.
4. This will launch a wizard used to deploy applications to Bluemix.
5. In the **Name** field, you can change the name of the application.
6. The **Subdomain** will be part of the URL of the application so it must be unique, in the **Subdomain** field add your name or a random number at the end, ie: **CognitiveApp79372**, we need to add this to avoid possible conflicts with other existing applications, click **Next**.
7. For this specific mobile application, we don't need any Bluemix services, so keep the defaults and click **Next**.
8. For this specific mobile application, we don't need any environment variable, so keep the defaults and click **Finish**.
9. Wait for the application to be deployed, the deployment time depends on the internet speed.

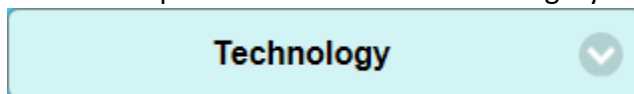
Once the application is deployed, the **Internal Web Browser** will be opened, but now the application is running in the cloud, meaning that it can be accessed from any device with a web browser, to test it you can open a web browser in your smartphone and enter the URL.

3.3.7 Test the solution

1. Use the buttons at the bottom of the web page to switch to the Setting page:

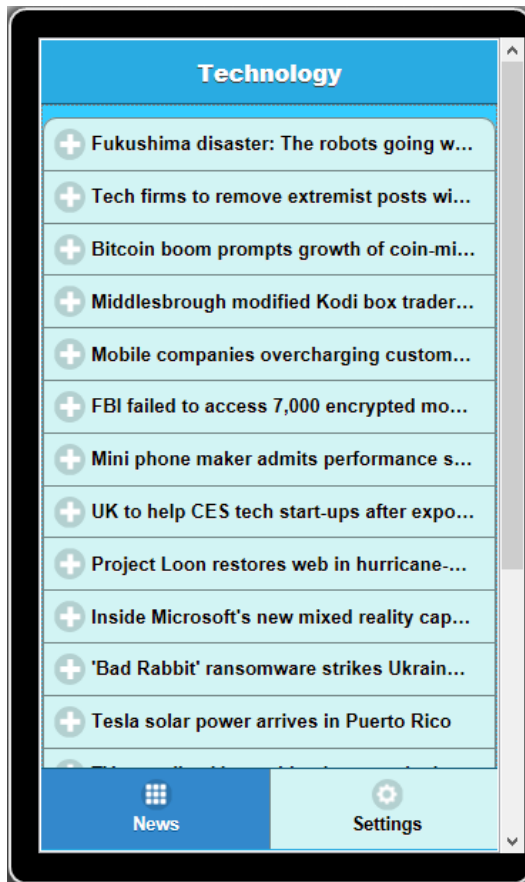


2. Use the drop-down menu to select a category of news:



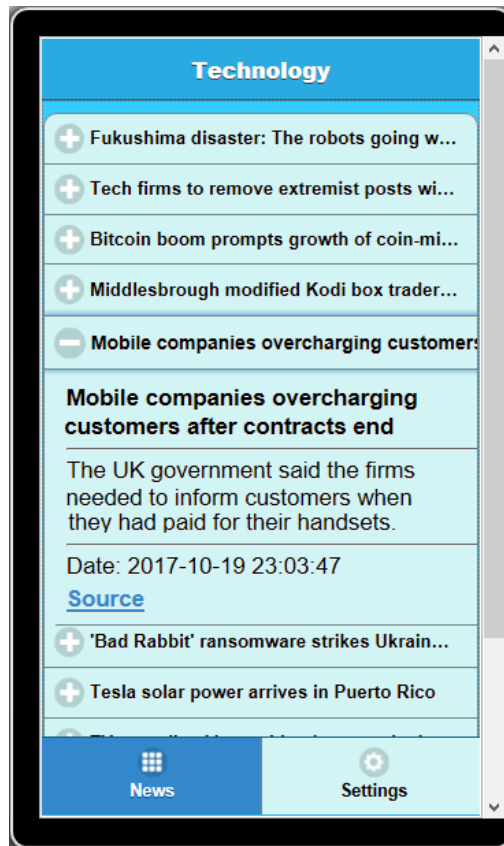
CASCON 2017

3. Use the buttons at the bottom of the web page to switch to the News page, now it will load news based on the selection you made:



CASCON 2017

- Click on the + icon on any news to expand it to see more details as well as a link to the source:



- Access your Twitter account and post a tweet, the colors of the application will be automatically adjusted to match the tone of your post.

The **Watson Tone Analyzer** service is capable of recognizing 5 tones, anger, disgust, fear, joy and sadness, depending on the tone of your post, one of the following colors will be automatically used:

Tone	Color
Anger	Blue
Disgust	Brown
Fear	Orange
Joy	Yellow
Sadness	Red

To test the IoT integration simply cover the device to prevent light from reaching the sensor, you will see how the brightness of the application is automatically adjusted, you can do this gradually too to allow more or less light from reaching the sensor and the brightness will be adjusted accordingly.

CASCON 2017

Notice that in the **Settings** page, you can change how often the application is refreshed, default is 10 seconds.