{ Lab 3 }

IoT application in IBM Cloud with Node-RED and IBM Watson

May 2018 - v2.1

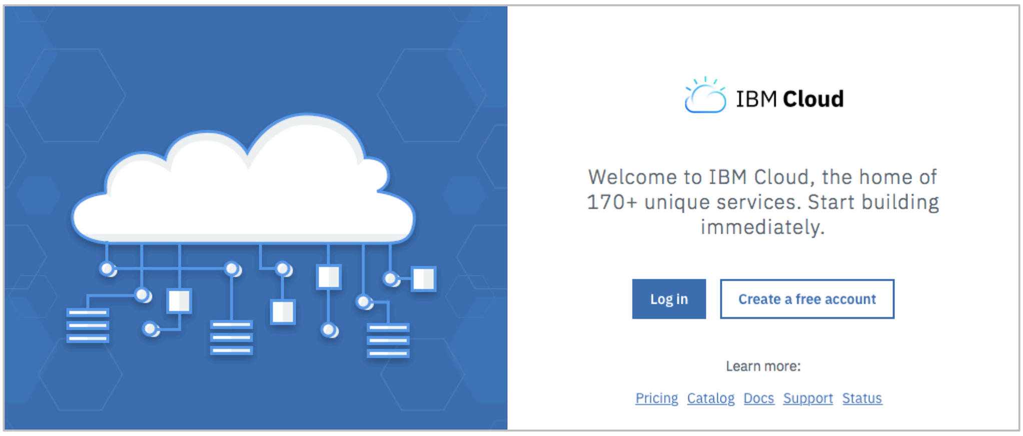
Agenda

Before Starting

Information

This hands-on required to have an IBM Cloud account. If you don't, you can create one here: http://bluemix.net/.

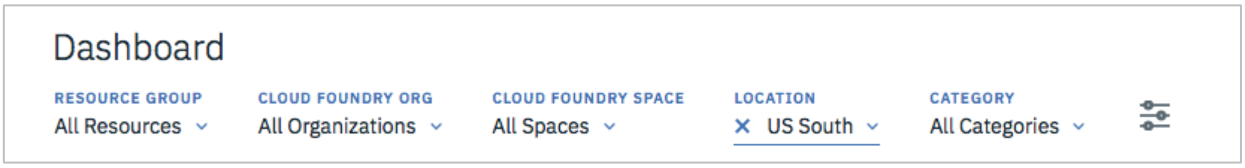
- Open a browser and access to IBM Cloud: https://console.bluemix.net.
- If you have an IBM Cloud account, click **Log in**, and enter your IBM ID credentials. If you don't have an IBM Cloud account, click **Create a free account**. Enter your email address, and additional information required. You will receive an email with activation link. Once activated, you could use your new free IBM cloud account: log in.



IBM Cloud

Welcome to IBM Cloud, the home of 170+ unique services. Start building immediately.

- Select organization, location and space to use during this lab.



Dashboard

RESOURCE GROUP	CLOUD FOUNDRY ORG	CLOUD FOUNDRY SPACE	LOCATION	CATEGORY
All Resources	All Organizations	All Spaces	US South	All Categories

- If needed, free resources (GB / #Services) in your IBM Cloud Organization & Spaces to run the lab exercises. If you encounter a resource contention (error message saying you are out of resources), clean up your spaces by deleting existing Apps or Services.

1. Hands-on presentation

Section 1. Overview

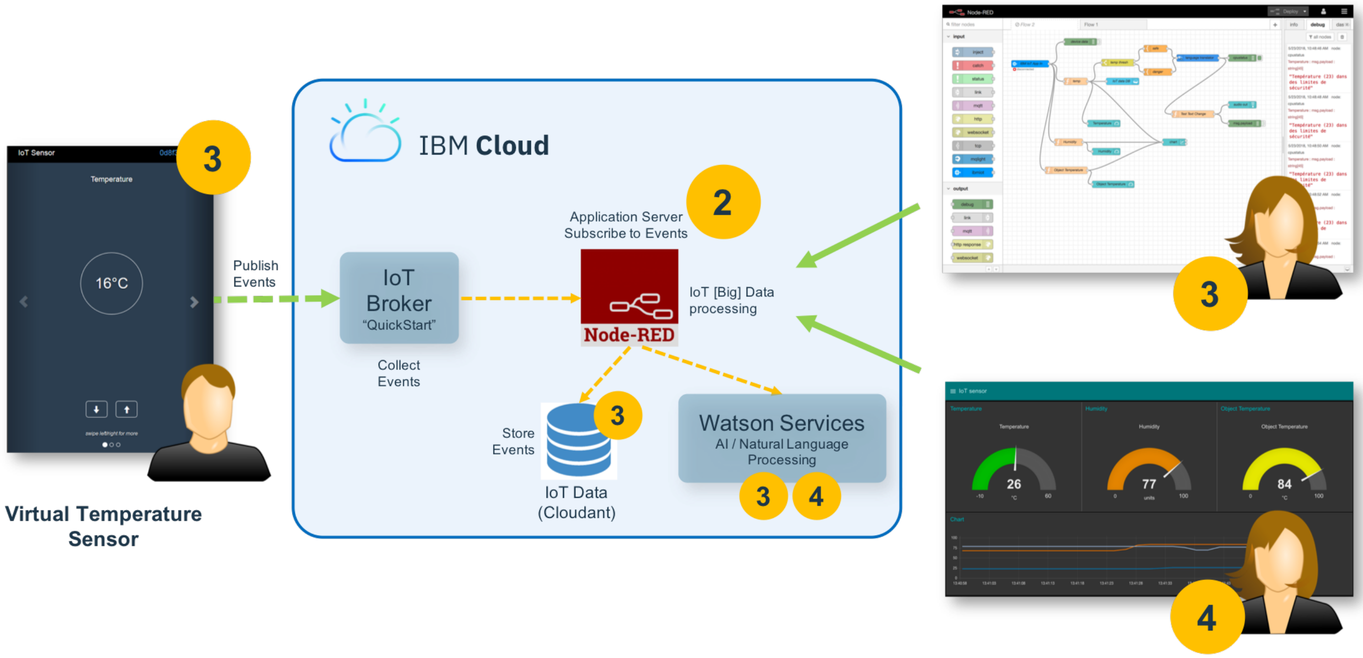
In this hands-on session, you will create a Node-RED application in IBM Cloud to collect, store and display virtual sensor data.

Node-RED (https://nodered.org/) is a flow-based programming tool, originally developed by the IBM Emerging Technology Services team (in early 2013) and now a part of JS Foundation. Traditional development can be very technical, but Node-RED enables you to concentrate on the logic of your workflow and allows fast prototyping.
Node-RED consists of a Node.js-based runtime with a flow editor accessed through a web browser. Within the browser, you create your application by dragging nodes from a customizable palette into a workspace and start to wire them together. With a single click, the application is deployed back to its runtime.

Session objectives are:

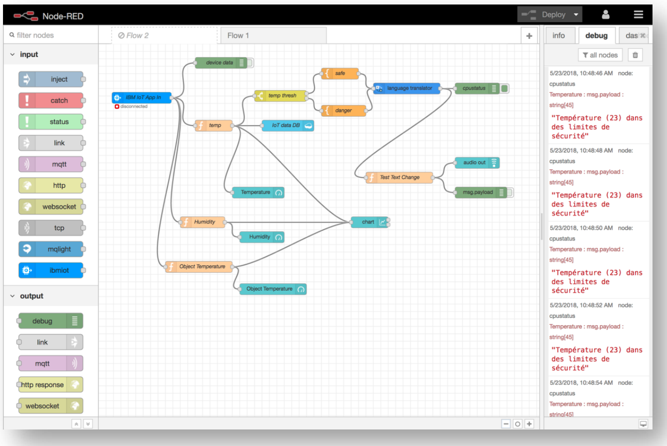
- Create & modify an application using Node-RED in IBM Cloud
- Discover new services & Node-RED to consume or create services (IoT / database…)
- Discover Watson services

Find below lab overview (with exercise numbers).



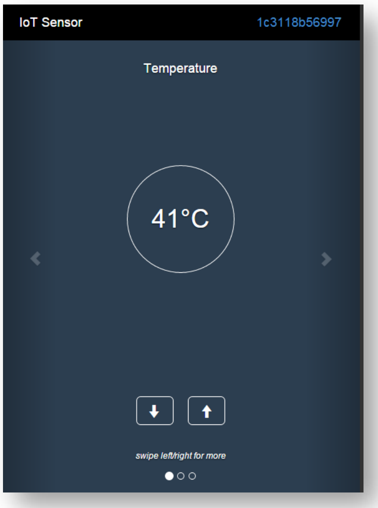
Expected results are :

- Your Node-RED application is operational (using Node.js runtime), accessing Cloudant & IoT platform (QuickStart)

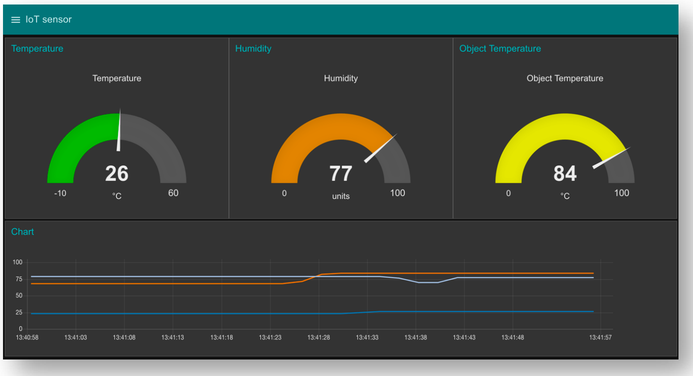


```
5/23/2018, 10:48:46 AM  node: cpustatus
Temperature : msg.payload : string[45]
"Température (23) dans des limites de sécurité"
5/23/2018, 10:48:48 AM  node: cpustatus
Temperature : msg.payload : string[45]
"Température (23) dans des limites de sécurité"
5/23/2018, 10:48:50 AM  node: cpustatus
Temperature : msg.payload : string[45]
"Température (23) dans des limites de sécurité"
5/23/2018, 10:48:52 AM  node: cpustatus
Temperature : msg.payload : string[45]
"Température (23) dans des limites de sécurité"
5/23/2018, 10:48:54 AM  node: cpustatus
Temperature : msg.payload : string[45]
```

- Your Node-RED app is online (reachable from the Internet), & will be connected to a temperature simulator (sensor)



- Optionally, you are able to provide a dashboard with voice alert.



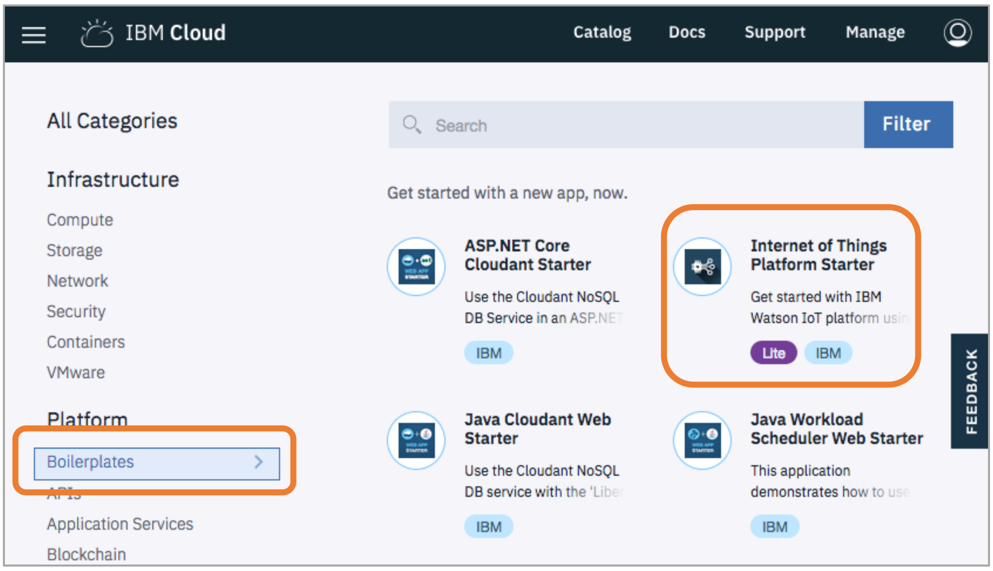
Section 2. Prerequisites

The required file used during this lab can be downloaded the JSON files from

- http://ibmcloud-watson-day.mybluemix.net/files/Lab3_IoT_Flow.v2.1.json
- http://ibmcloud-watson-day.mybluemix.net/files/Lab3_IoT_Dashboard.v2.1.json

2. Create Node-RED application and Login

__ 1. In IBM Cloud Catalog, choose **Boilerplate** category



Get started with a new app, now.

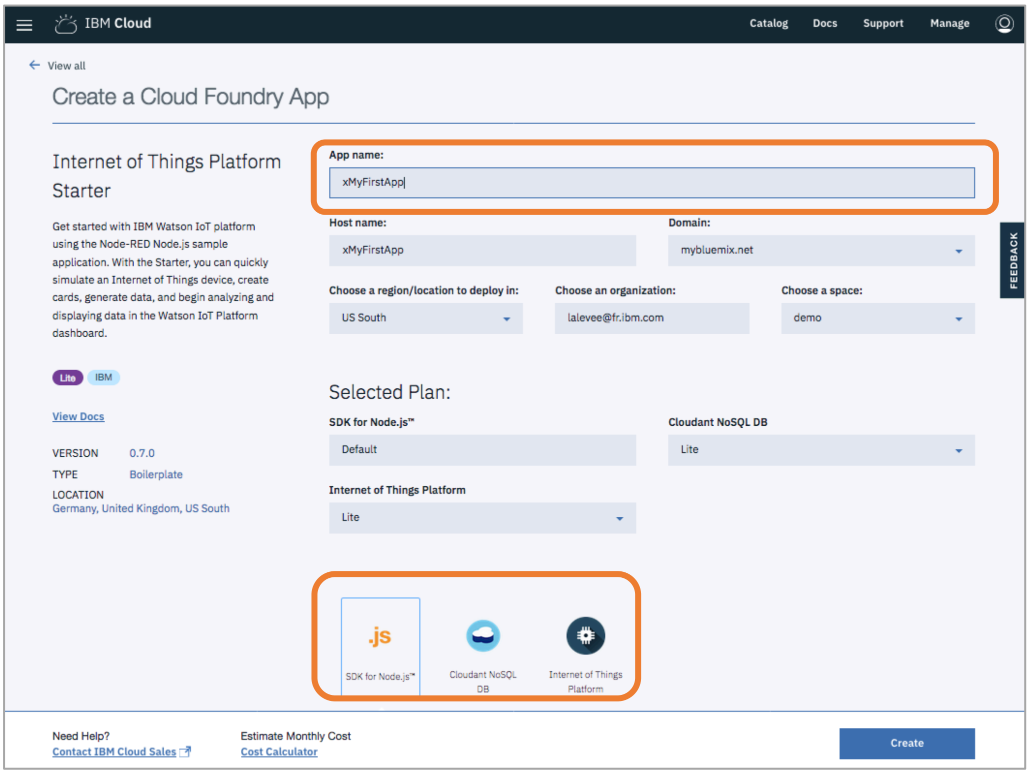
ASP.NET Core Cloudant Starter
Use the Cloudant NoSQL DB Service in an ASP.NET

Internet of Things Platform Starter
Get started with IBM Watson IoT platform usin

Java Cloudant Web Starter
Use the Cloudant NoSQL DB service with the 'Liber

Java Workload Scheduler Web Starter
This application demonstrates how to use

__ 2. Click on **Internet of Things Platform Starter** to create an instance:
Fill in the App Name & Host name fields.
Note: Node-RED is a Node.js based application: using this boilerplate will instantiate a Node.js runtime + a Cloudant (NoSQL DB) service.



Create a Cloud Foundry App

Internet of Things Platform Starter

Get started with IBM Watson IoT platform using the Node-RED Node.js sample application. With the Starter, you can quickly simulate an Internet of Things device, create cards, generate data, and begin analyzing and displaying data in the Watson IoT Platform dashboard.

VERSION 0.7.0
TYPE Boilerplate
LOCATION Germany, United Kingdom, US South

App name: xMyFirstApp

Host name: xMyFirstApp

Domain: mybluemix.net

Choose a region/location to deploy in: US South

Choose an organization: lalevee@fr.ibm.com

Choose a space: demo

Selected Plan:

SDK for Node.js™: Default

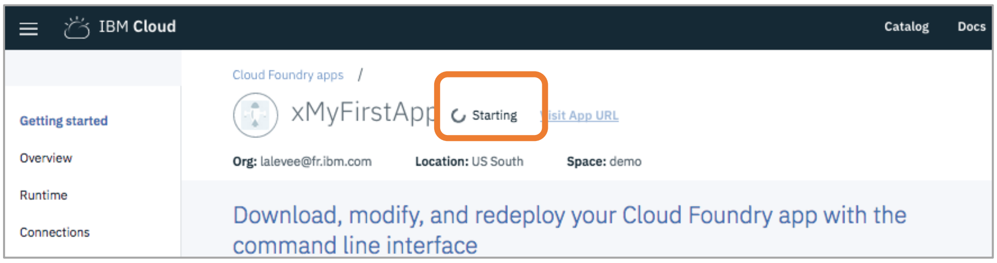
Cloudant NoSQL DB: Lite

Internet of Things Platform: Lite

Need Help?
Contact IBM Cloud Sales

Estimate Monthly Cost
Cost Calculator

__ 3. Click **Create**.
Wait for the environment to be created & the app to start (~4 minutes).

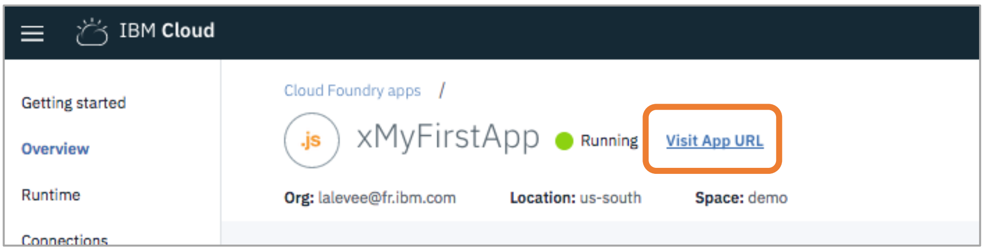


xMyFirstApp Starting Visit App URL

Org: lalevee@fr.ibm.com **Location:** US South **Space:** demo

Download, modify, and redeploy your Cloud Foundry app with the command line interface

__ 4. By clicking **Visit App URL**, access the Node-RED application

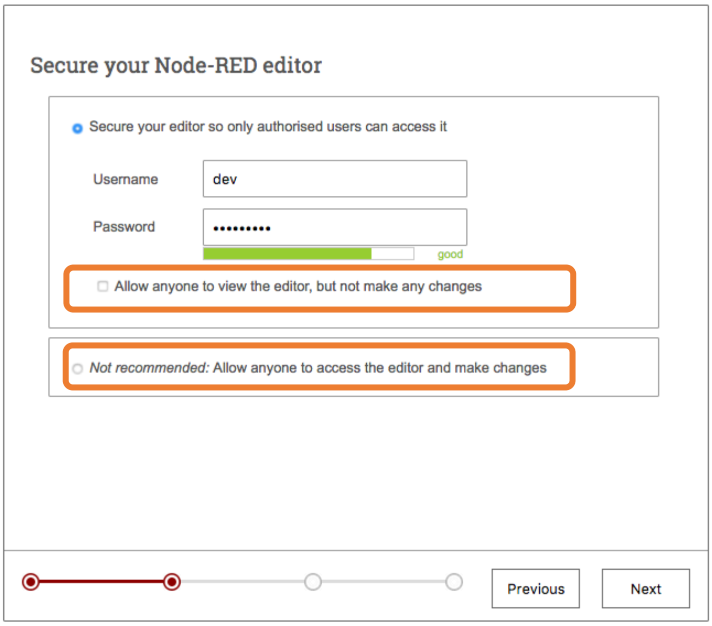


xMyFirstApp ● Running Visit App URL

Org: lalevee@fr.ibm.com **Location:** us-south **Space:** demo

__ 5. Run the wizard to configure authentication: secure your editor with your own credentials so only authorized users can access it (Node-RED has its own authentication system).

Don't check **Allow anyone to view the editor, but not make any changes** and **Allow anyone to view the editor**.



Secure your Node-RED editor

● Secure your editor so only authorised users can access it

Username: dev

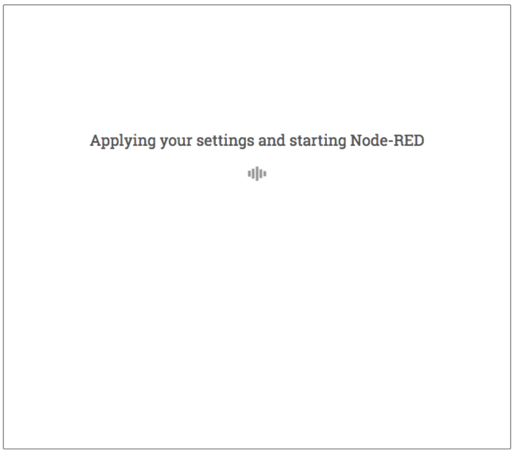
Password: •••••••••

good

□ Allow anyone to view the editor, but not make any changes

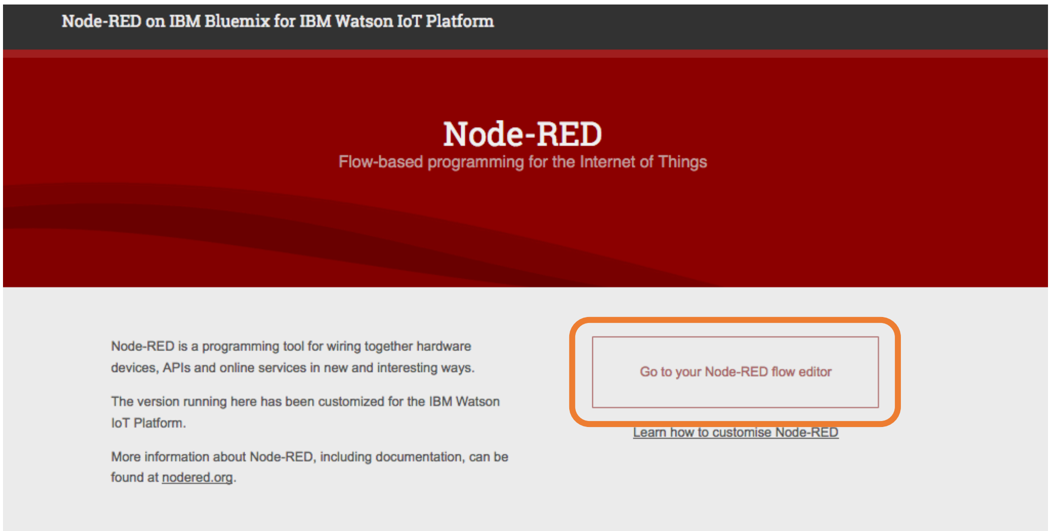
○ *Not recommended:* Allow anyone to access the editor and make changes

__ 6. On wizard last step screen, click on **Finish** to start Node-RED.



Applying your settings and starting Node-RED

__ 7. Node-RED is a browser-based editor that makes it easy to wire together flows that can be deployed to the runtime. In the case of IoT, Node-RED is really powerful to quickly test all the possibilities that IBM Cloud offers with different kind of services.
Your Node-RED app has a public URL like any web app (you defined it in step 2).
Click on **Go to your Node-RED flow editor** and use the credentials provided before.



Node-RED on IBM Bluemix for IBM Watson IoT Platform

Node-RED

Flow-based programming for the Internet of Things

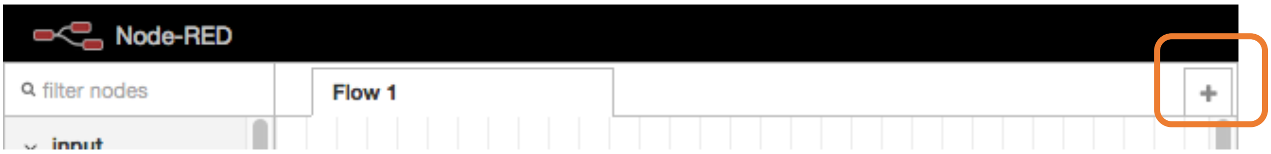
Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

The version running here has been customized for the IBM Watson IoT Platform.

More information about Node-RED, including documentation, can be found at nodered.org.

Go to your Node-RED flow editor
Learn how to customise Node-RED

__ 8. You now have access to the Node-RED UI.
Keep the existing default flow and create a new flow: click **+**. You will use it in next exercise.



3. Create sensor and a new flow

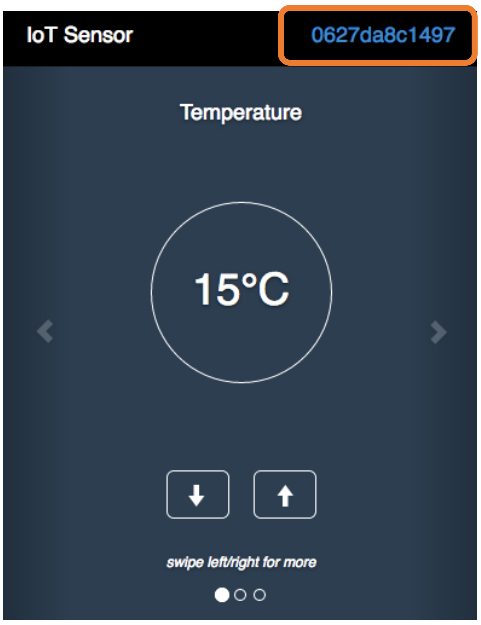
Section 1. Sensors & IoT

__ 1. Open a new window or tab in your browser.

__ 2. To create a sensor simulator, connect to http://ibm.biz/iotsensor
There are 3 simulated sensors:

- Object temperature
- Temperature
- Humidity

The simulator (from IBM Cloud IoT Quickstart) connects automatically and starts publishing data.
It must remain connected to visualize the data.
Use the simulator buttons to change the simulated sensor readings. Data is published periodically.



Note: Instead of using your desktop browser, you can use your smartphone.

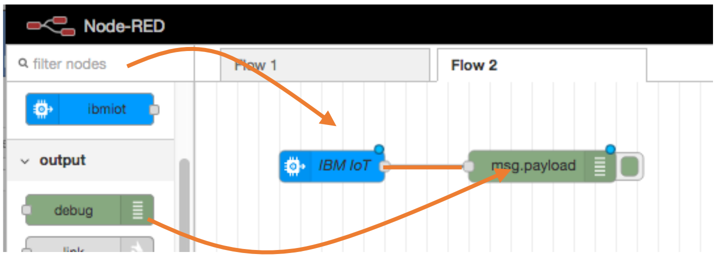
__ 3. Identify your virtual device ID (top right corner) : copy it. You will use it in next section.
Warning: if you reload this page, the device ID changes.

Section 2. Node-RED flow: creation & importation

__ 1. Go back to Node-RED window

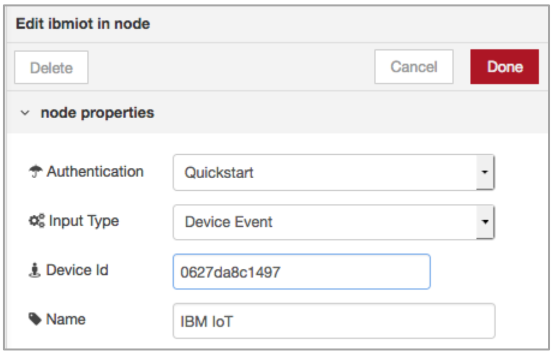
__ 2. From left panel, drag and drop nodes to the workspace

- Chose the Input node **ibmiot**
- Add an output **debug** node
- Link them



__ 3. Configure IBM IoT by double clicking on it :

- Authentication: Quickstart (means it is a simple authentication – for demo purposes)
- Device ID : <The value from Section 1, step 3 - Generated by the Simulator>



Authentication: Quickstart

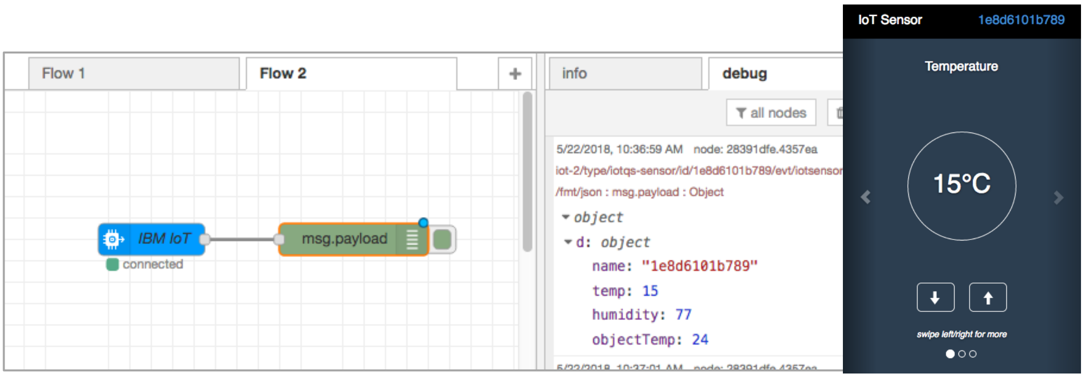
Input Type: Device Event

Device Id: 0627da8c1497

Name: IBM IoT

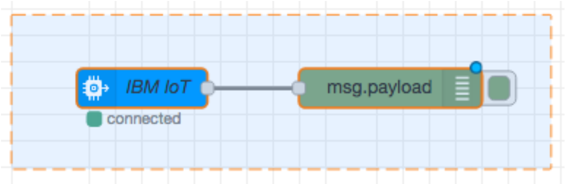
__ 4. Click **Done** & deploy your flow by clicking the **Deploy** button (top right).

__ 5. Check the **Debug** Panel on the right side while you are playing with the sensor simulator. You should receive Device (sensor = web app. you opened in other window) data as the IBM IoT Node subscribed to this particular Device topic.

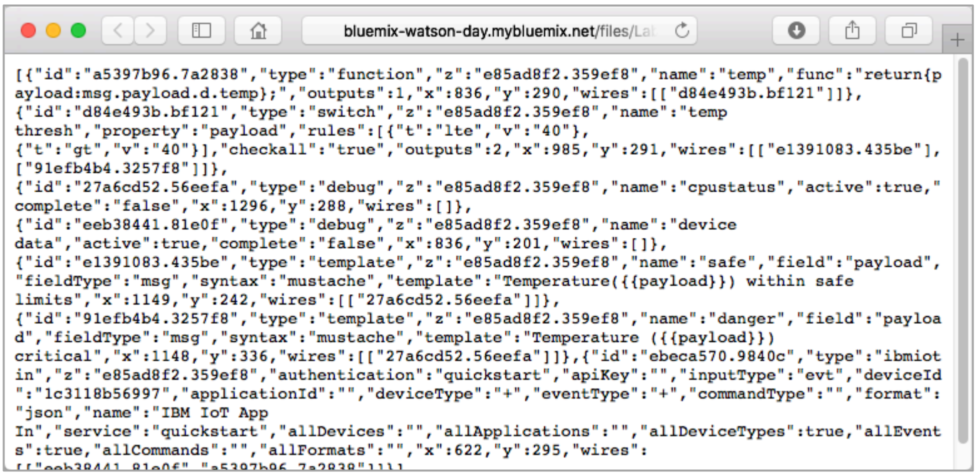


```
5/22/2018, 10:36:59 AM   node: 28391dfe.4357ea
iot-2/type/iotqs-sensor/id/1e8d6101b789/evt/iotsensor
/fmt/json : msg.payload : Object
▾object
 ▾d: object
    name: "1e8d6101b789"
    temp: 15
    humidity: 77
    objectTemp: 24
5/22/2018, 10:37:01 AM   node: 28391dfe.4357ea
```

__ 6. Delete the whole flow by selecting all the nodes & pressing the 'Delete' key.



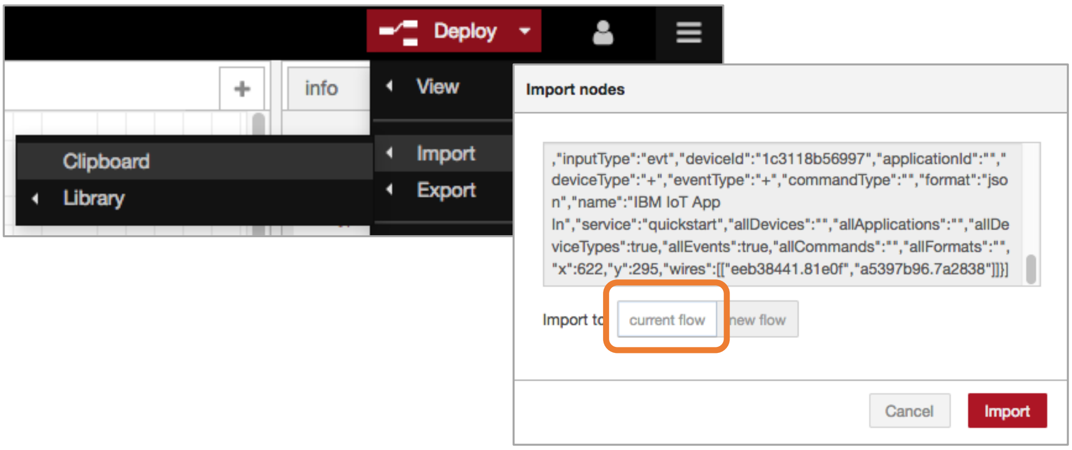
__ 7. Now import a new flow. A flow can be exported and imported using JSON file.
Open link http://ibmcloud-watson-day.mybluemix.net/files/Lab3_IoT_Flow.v2.1.json to display code in JSON format. You can also open the file you downloaded previously.



```
[{"id":"a5397b96.7a2838","type":"function","z":"e85ad8f2.359ef8","name":"temp","func":"return{p
ayload:msg.payload.d.temp};","outputs":1,"x":836,"y":290,"wires":[["d84e493b.bf121"]]},
{"id":"d84e493b.bf121","type":"switch","z":"e85ad8f2.359ef8","name":"temp
thresh","property":"payload","rules":[{"t":"lte","v":"40"},
{"t":"gt","v":"40"}],"checkall":"true","outputs":2,"x":985,"y":291,"wires":[["e1391083.435be"],
["91efb4b4.3257f8"]]},
{"id":"27a6cd52.56eefa","type":"debug","z":"e85ad8f2.359ef8","name":"cpustatus","active":true,"
complete":"false","x":1296,"y":288,"wires":[]},
{"id":"eeb38441.81e0f","type":"debug","z":"e85ad8f2.359ef8","name":"device
data","active":true,"complete":"false","x":836,"y":201,"wires":[]},
{"id":"e1391083.435be","type":"template","z":"e85ad8f2.359ef8","name":"safe","field":"payload",
"fieldType":"msg","syntax":"mustache","template":"Temperature({{payload}}) within safe
limits","x":1149,"y":242,"wires":[["27a6cd52.56eefa"]]},
{"id":"91efb4b4.3257f8","type":"template","z":"e85ad8f2.359ef8","name":"danger","field":"payloa
d","fieldType":"msg","syntax":"mustache","template":"Temperature ({{payload}})
critical","x":1148,"y":336,"wires":[["27a6cd52.56eefa"]]},{"id":"ebeca570.9840c","type":"ibmiot
in","z":"e85ad8f2.359ef8","authentication":"quickstart","apiKey":"","inputType":"evt","deviceId
":"1c3118b56997","applicationId":"","deviceType":"+","eventType":"+","commandType":"","format":
"json","name":"IBM IoT App
In","service":"quickstart","allDevices":"","allApplications":"","allDeviceTypes":true,"allEvent
s":true,"allCommands":"","allFormats":"","x":622,"y":295,"wires":
[["eeb38441.81e0f","a5397b96.7a2838"]]}]
```

__ 8. Select all and copy it

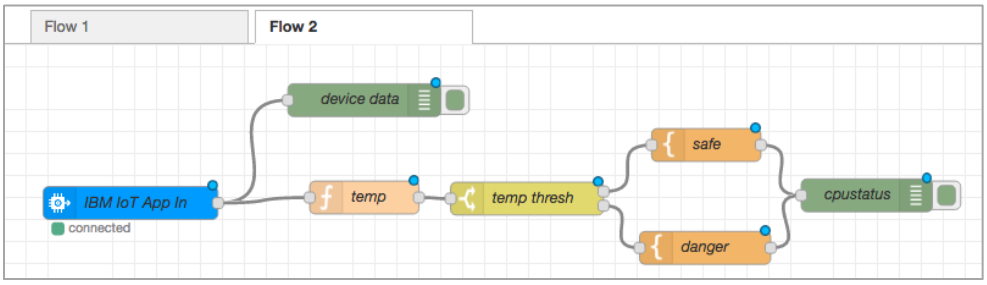
__ 9. Click on the top right button near **Deploy**.
Select **Import**, **Clipboard** & copy/paste the content of the JSON file in **Current flow**.



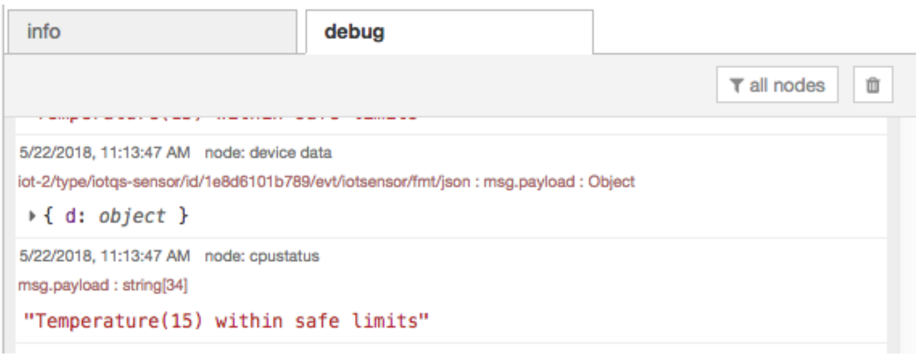
```
,"inputType":"evt","deviceId":"1c3118b56997","applicationId":"","
deviceType":"+","eventType":"+","commandType":"","format":"jso
n","name":"IBM IoT App
In","service":"quickstart","allDevices":"","allApplications":"","allDe
viceTypes":true,"allEvents":true,"allCommands":"","allFormats":"",
"x":622,"y":295,"wires":[["eeb38441.81e0f","a5397b96.7a2838"]]}]
```

__ 10. Click on workspace to paste imported nodes

__ 11. Fill in the **Device ID** field in the **IBM IoT App In** node.



__ 12. Click **Deploy** to deploy the new Flow.
Modify the Device Temperature & check the **Debug** logs.

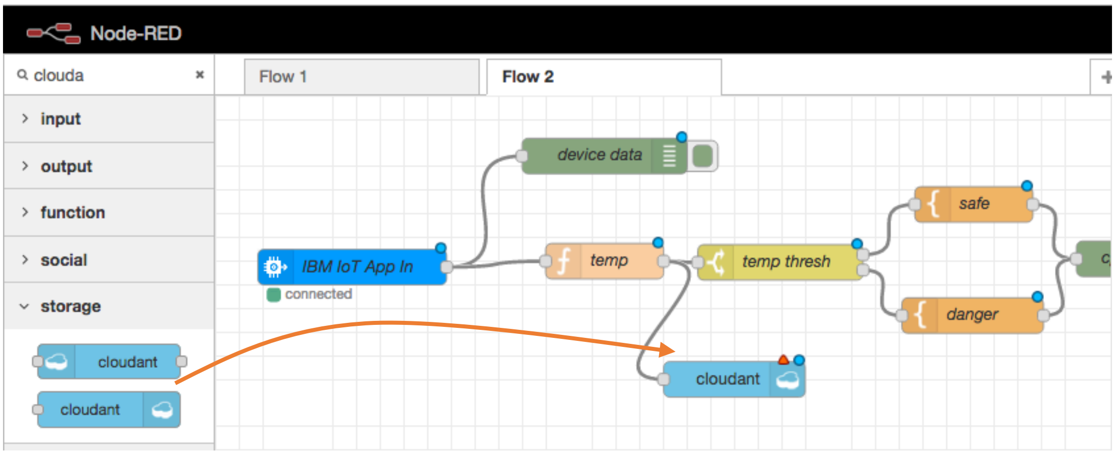


```
5/22/2018, 11:13:47 AM   node: device data
iot-2/type/iotqs-sensor/id/1e8d6101b789/evt/iotsensor/fmt/json : msg.payload : Object
▸{ d: object }
5/22/2018, 11:13:47 AM   node: cpustatus
msg.payload : string[34]
"Temperature(15) within safe limits"
```

Section 3. Insert IoT Data in Cloudant DB

Let's insert the event data coming from the Device sensors in a Cloudant database! Remember that you already have a Cloudant service deployed for Node-RED. You will use it to store your data.

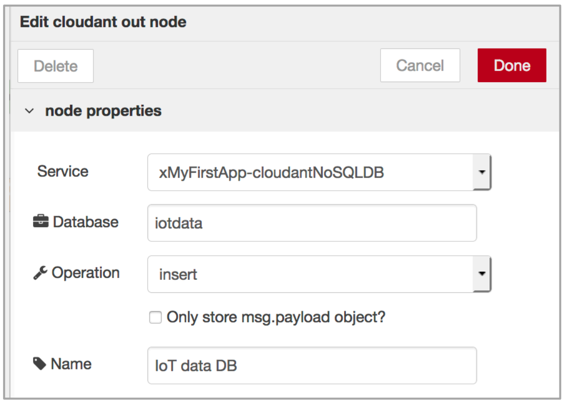
__ 1. Add a Cloudant Node (**Cloudant OUT** node in the **Storage** Category) & link it to the **temp** function node



__ 2. Configure it:

- Service : Cloudant service name bound to your Node.js runtime. *As Node.js is already bound to a Cloudant Service, the service name should appear in the Drop-down list.*
- Database: name of your choice (lower case)
- Name (node): name of your choice

Click **Done**.


Edit cloudant out node

Service: xMyFirstApp-cloudantNoSQLDB

Database: iotdata

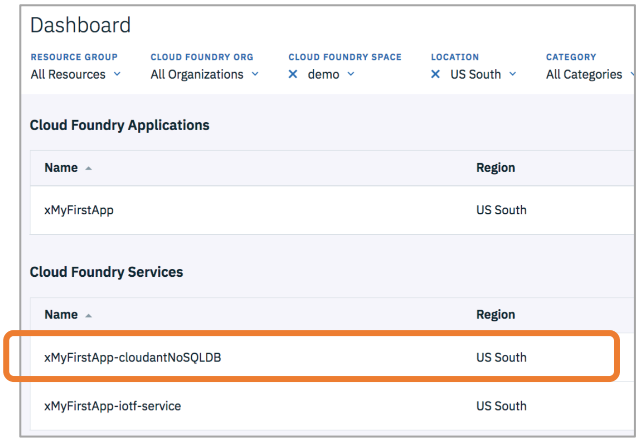
Operation: insert

□ Only store msg.payload object?

Name: IoT data DB

__ 3. Deploy your new flow (**Deploy** button)

__ 4. From your IBM Cloud Dashboard (IBM Cloud window of your browser), start the Cloudant dashboard by clicking on the line of Cloudant service



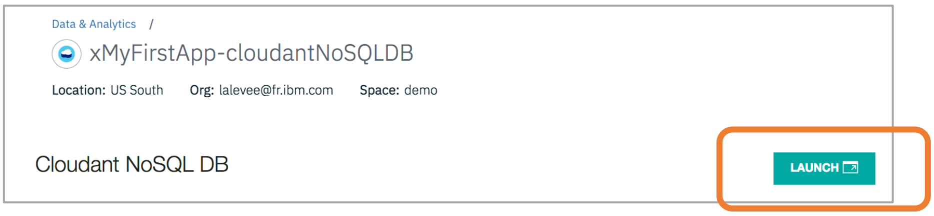
Cloud Foundry Applications

Name	Region
xMyFirstApp	US South

Cloud Foundry Services

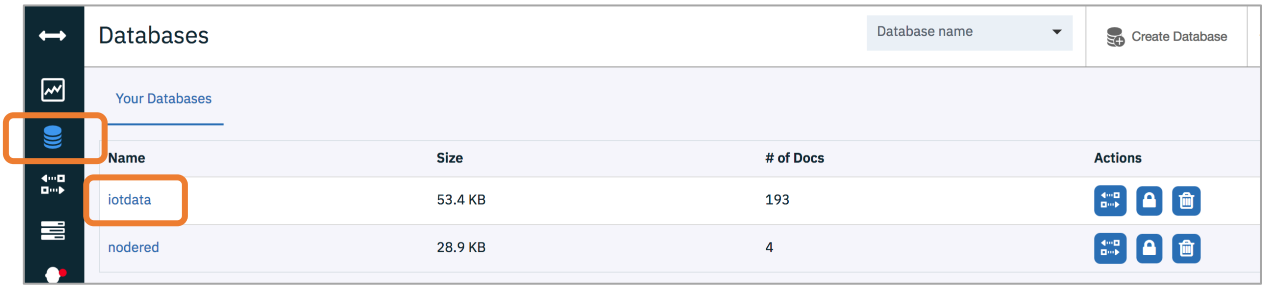
Name	Region
xMyFirstApp-cloudantNoSQLDB	US South
xMyFirstApp-iotf-service	US South

__ 5. Click **Launch** button to start Cloudant console



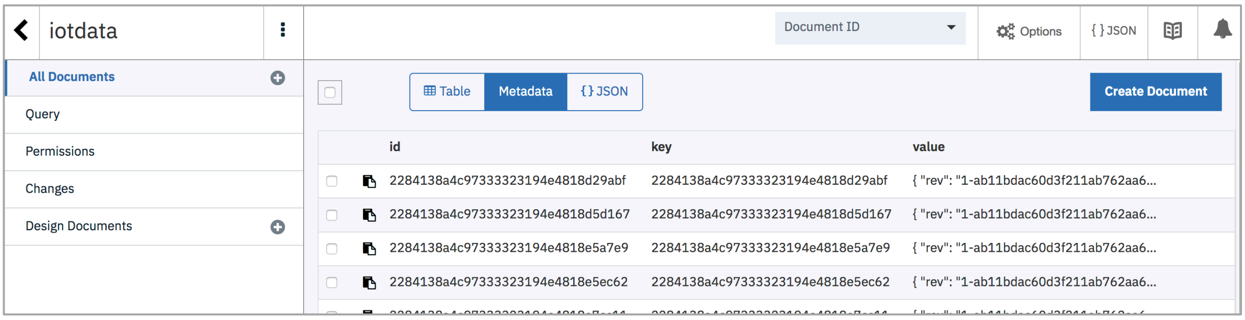
Location: US South Org: lalevee@fr.ibm.com Space: demo

__ 6. Select **Database** icon in left panel, then your database name (defined in step 2).



Name	Size	# of Docs	Actions
iotdata	53.4 KB	193	
nodered	28.9 KB	4	

__ 7. Have a look to the inserted data in the database. Click on record to see content.

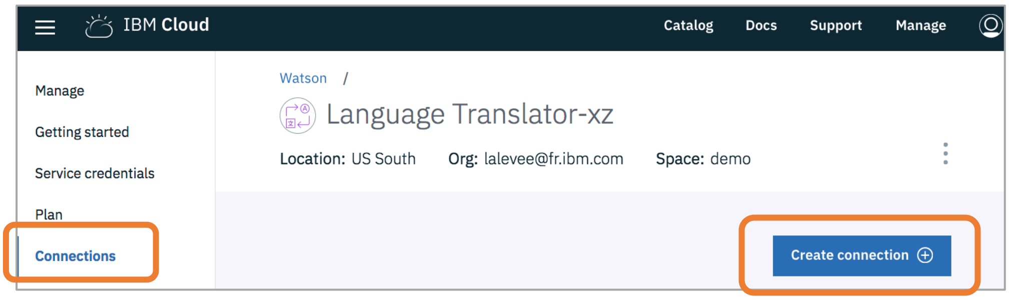


	id	key	value
□	2284138a4c97333323194e4818d29abf	2284138a4c97333323194e4818d29abf	{ "rev": "1-ab11bdac60d3f211ab762aa6...
□	2284138a4c97333323194e4818d5d167	2284138a4c97333323194e4818d5d167	{ "rev": "1-ab11bdac60d3f211ab762aa6...
□	2284138a4c97333323194e4818e5a7e9	2284138a4c97333323194e4818e5a7e9	{ "rev": "1-ab11bdac60d3f211ab762aa6...
□	2284138a4c97333323194e4818e5ec62	2284138a4c97333323194e4818e5ec62	{ "rev": "1-ab11bdac60d3f211ab762aa6...

Section 4. Process IoT Data with Watson

__ 1. In IBM Cloud windows, add a « Watson Language Translator » service to your existing Node-RED application. From **Catalog**, click on **Language Translator** in Watson category.
Select **Lite** plan, then click **Create**. Service is deployed.

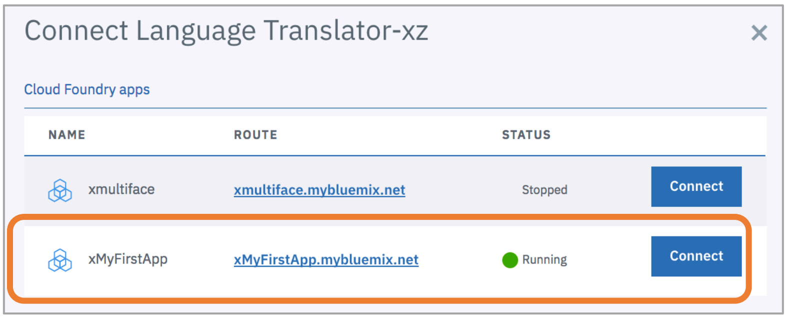
__ 2. From Language Translator dashboard, **Connections** menu, click on **Create connection**.



Language Translator-xz

Location: US South Org: lalevee@fr.ibm.com Space: demo

__ 3. Connect service to your Node-RED application



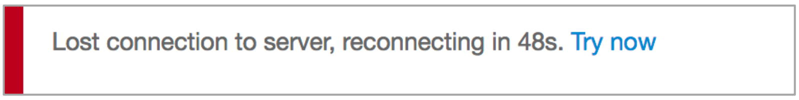
Connect Language Translator-xz

NAME	ROUTE	STATUS	
xmultiface	xmultiface.mybluemix.net	Stopped	Connect
xMyFirstApp	xMyFirstApp.mybluemix.net	● Running	Connect

__ 4. Accept the restage step to actually bind the service to the app.

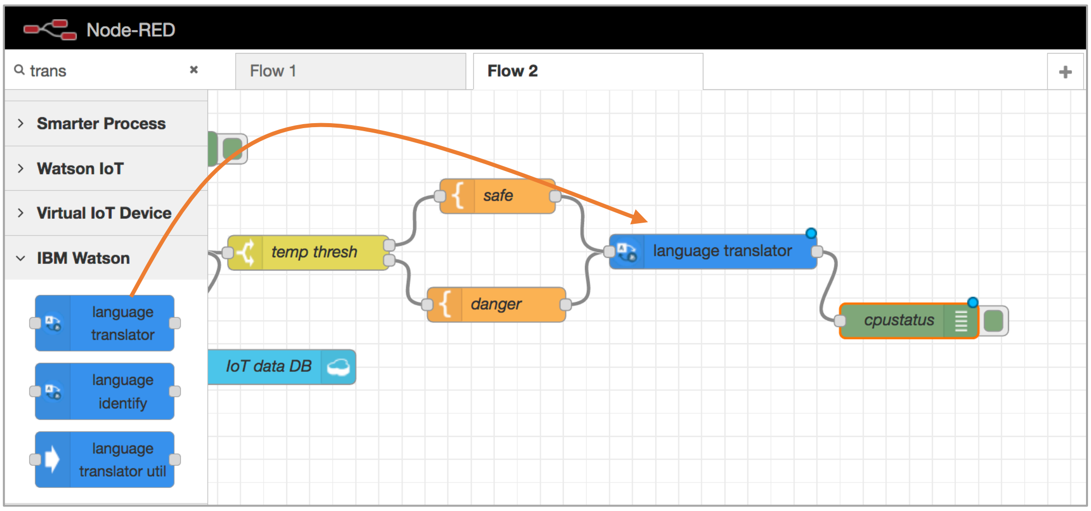
__ 5. While it is restaging (~3 minutes), take a look at **Service Credentials**. This information is useful if you want to invoke your Watson Service from any program (running in IBM Cloud or outside IBM Cloud)

__ 6. Go back to Node-RED window.
If you get a connection error message, your application restaging is not finished. Wait. Try to refresh page.



Lost connection to server, reconnecting in 48s. Try now

__ 7. Go back to the Node-RED environment.
Add **Language translator** node and link it between the template (*safe & danger) &* debug nodes (*cpustatus*).

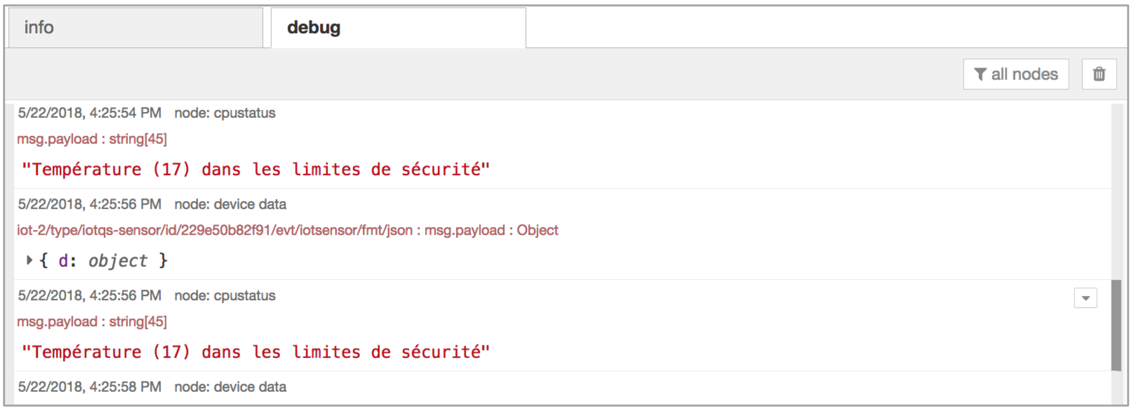


__ 8. Configure the Watson language translator node:

- Name (of your choice)
- Mode: **Translate**
- Domains: **Conversational**
- Source: **English**
- Target: **French** (or Spanish, Portuguese & Arabic)

Note: The user/password fields are not necessary & do not appear in the node settings if a Watson Language Translator service is properly bound to Node-RED application.

__ 9. **Deploy** your flow & check the logs (**debug** tab).

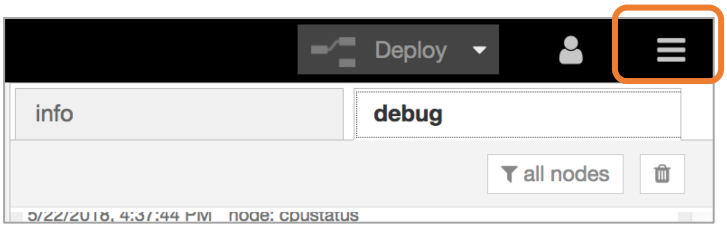


```
5/22/2018, 4:25:54 PM   node: cpustatus
msg.payload : string[45]
"Température (17) dans les limites de sécurité"
5/22/2018, 4:25:56 PM   node: device data
iot-2/type/iotqs-sensor/id/229e50b82f91/evt/iotsensor/fmt/json : msg.payload : Object
▸{ d: object }
5/22/2018, 4:25:56 PM   node: cpustatus
msg.payload : string[45]
"Température (17) dans les limites de sécurité"
5/22/2018, 4:25:58 PM   node: device data
```

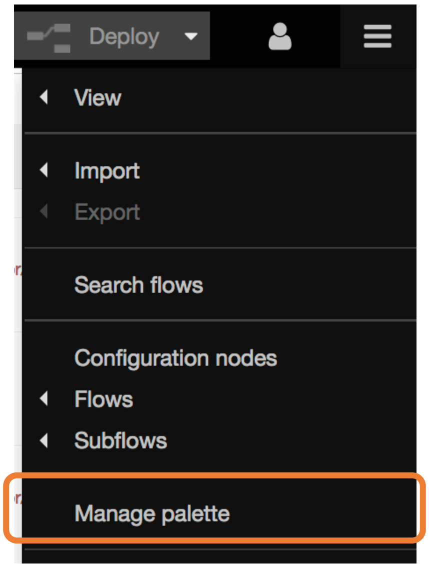
4. Create a dashboard application in Node-RED (optional)

Section 1. Import Node-RED Dashboarding capability

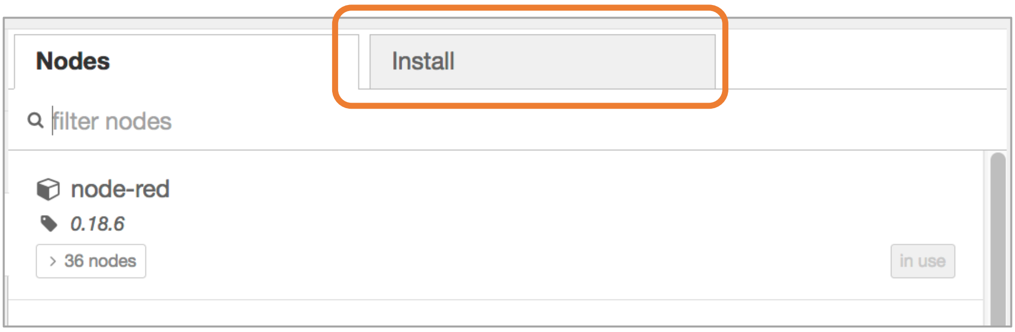
__ 1. At the top right-hand side of the page, click the ‘burger’ menu:



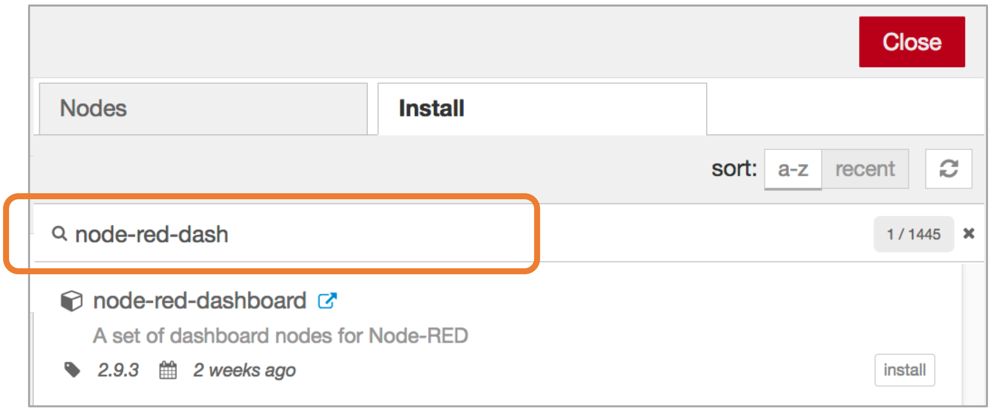
__ 2. Click **Manage palette**:



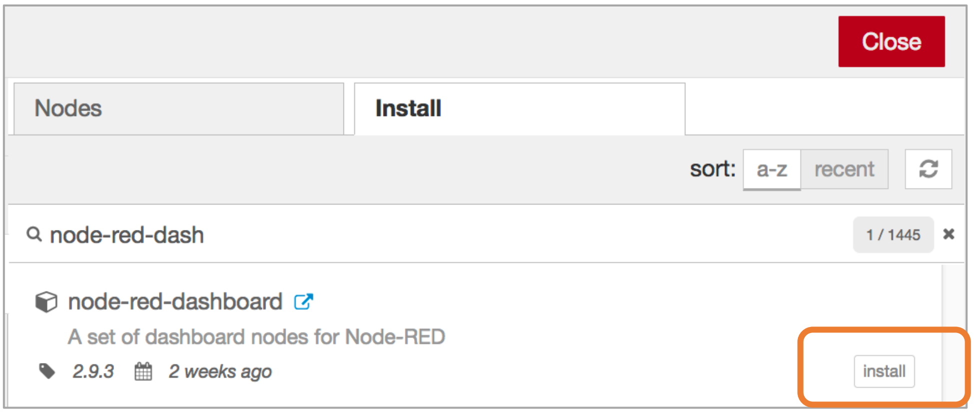
__ 3. In the sidebar that appears on the left-hand side of the page, click the Install tab:



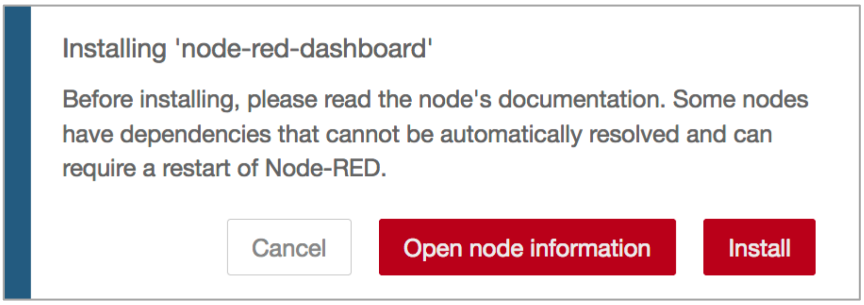
__ 4. In the search field, type **node-red-dash**:



__ 5. Next to the nod-red-dashboard result, click **install**



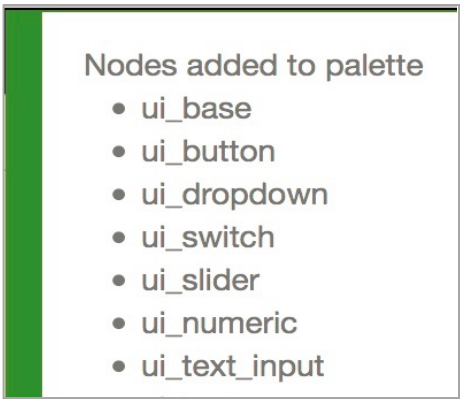
__ 6. Click **Install**



Installing 'node-red-dashboard'

Before installing, please read the node's documentation. Some nodes have dependencies that cannot be automatically resolved and can require a restart of Node-RED.

__ 7. Wait for the new nodes to be deployed.
A popup shows which nodes have been installed.

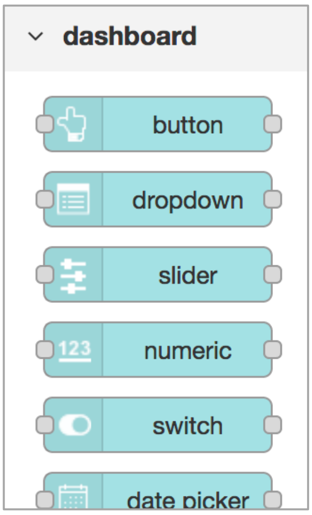


Nodes added to palette

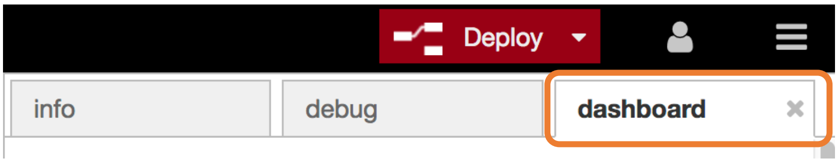
- ui_base
- ui_button
- ui_dropdown
- ui_switch
- ui_slider
- ui_numeric
- ui_text_input

__ 8. Click **Close**.

__ 9. Note the additional dashboard nodes on the palette : **Dashboard** category.



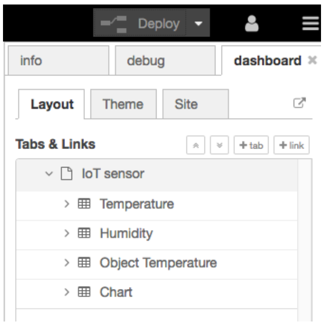
__ 10. Note also that there is a new dashboard tab in the right-hand sidebar:



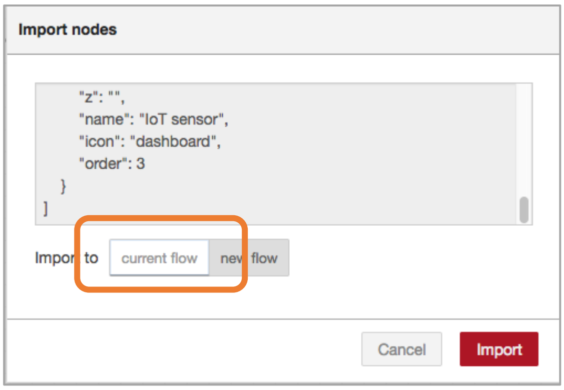
TIP: This dashboard tab may be used to add new tabs, menus etc. to the visualization dashboard. There are also two available themes – light and dark.

Section 2. Create a simple Node-RED Dashboard

In this section, you will create a simple dashboard for sensor data using new dashboard nodes installed.

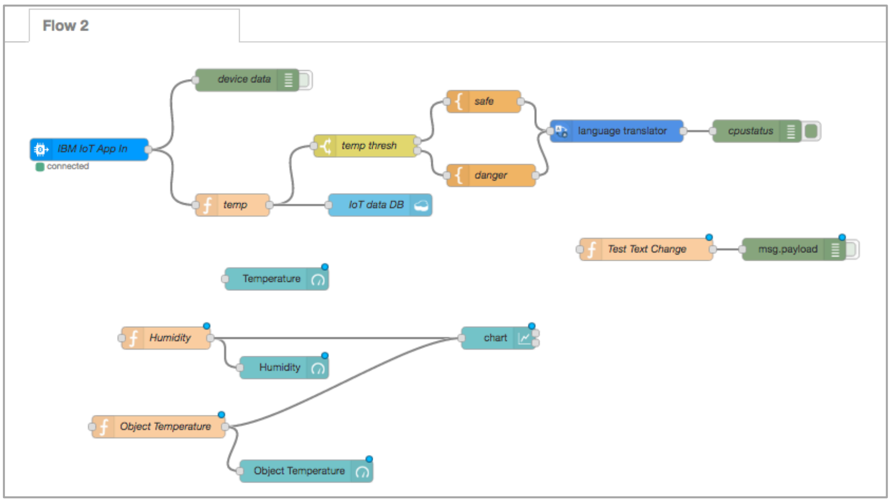


__ 1. In the current Node-RED tab, import the file named **Lab3_IoT_Dashboard.v2.0.txt** (previously downloaded): **Menu** > **Import** > **Clipboard**.
Click **current flow** then **Import**.



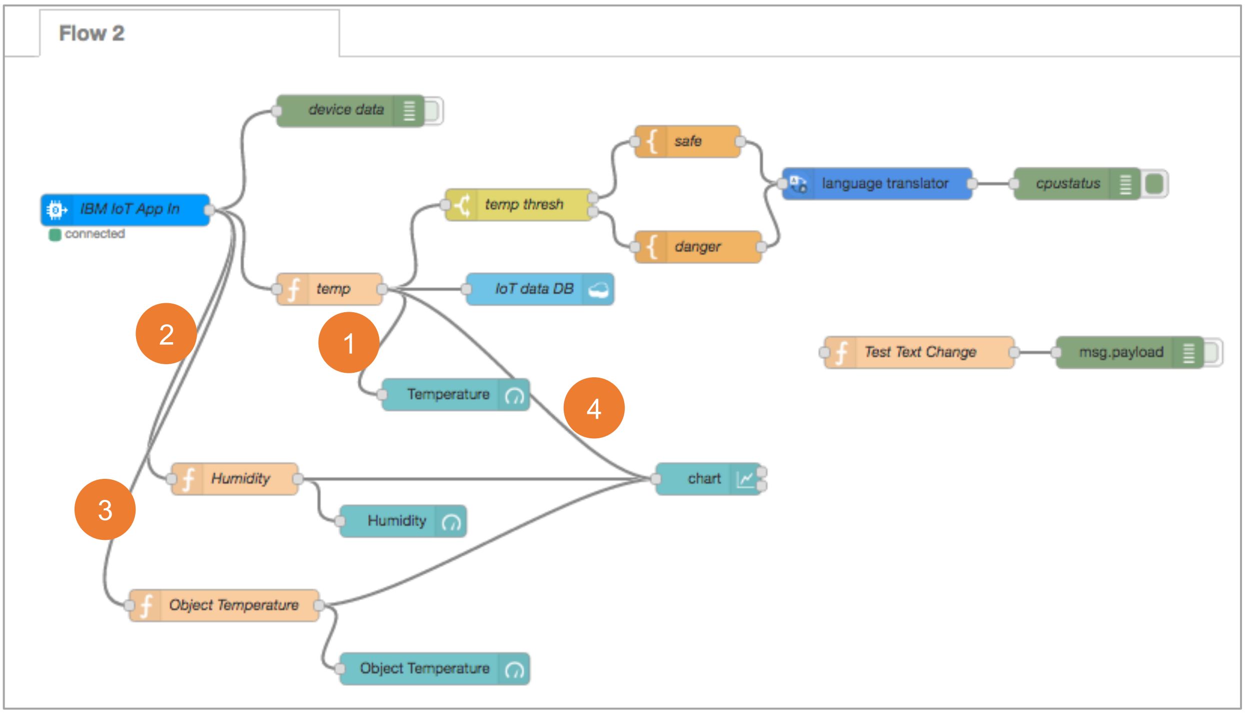
```
        "z": "",
        "name": "IoT sensor",
        "icon": "dashboard",
        "order": 3
    }
]
```

__ 2. Click on workspace to paste new nodes.



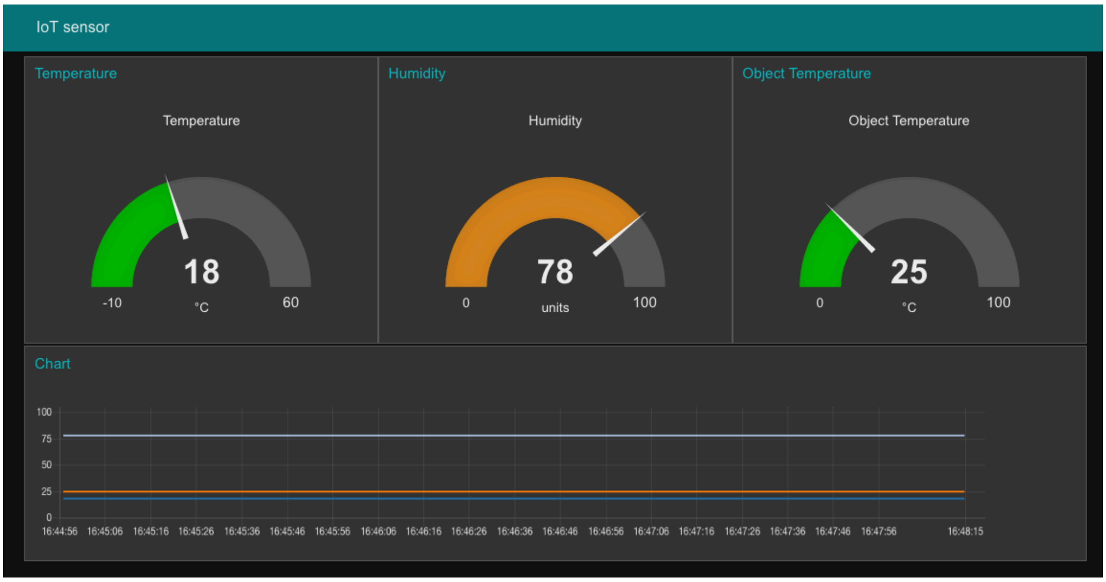
__ 3. Connect new nodes to existing…

1. **Temperature** to **temp**
2. **Humidity** to **IBM Iot App In**
3. **Object Temperature** to **IBM Iot App In**
4. **chart** to **temp**



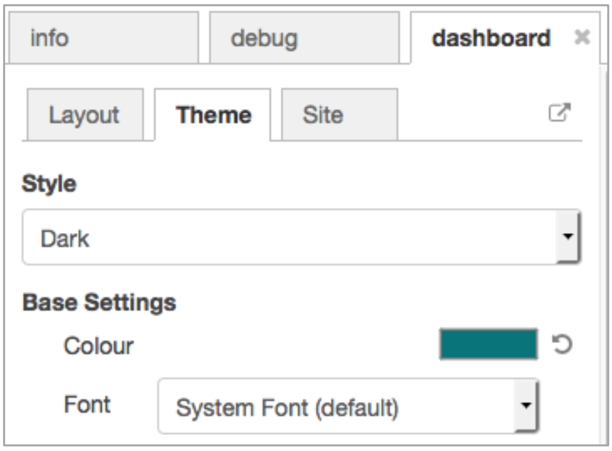
__ 4. Deploy the flow: **Deploy**

__ 5. Connect to **http://<YOUR_APP_HOSTNAME>/ui** (URL generated by Node-red-dashboard node) to see your new dashboard. Change value on virtual sensor app. to see impact on gauges and lines.



__ 6. Customize your dashboard using **Dashboard** tab

- **Theme**: use Dark style



Style

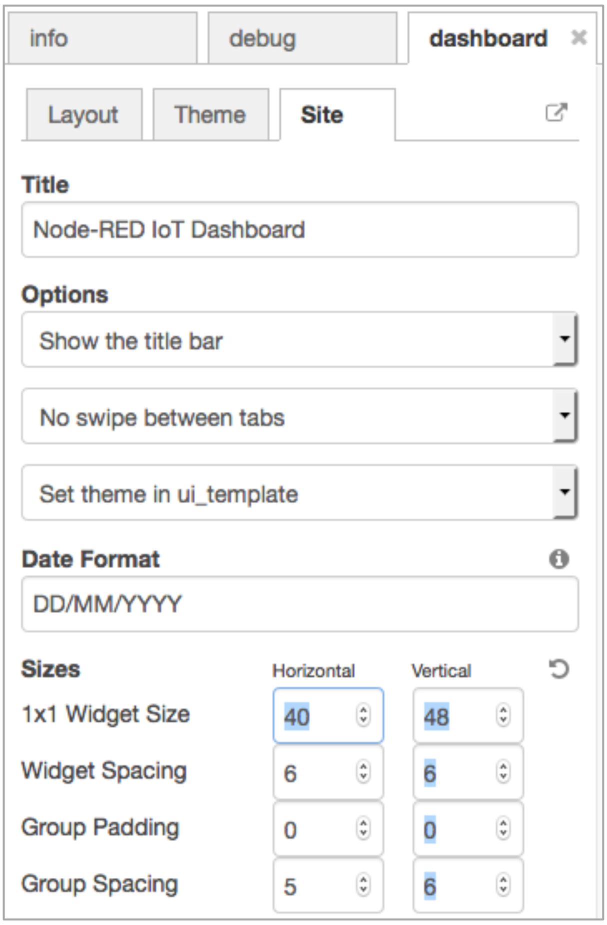
Dark

Base Settings

Colour 🟦

Font System Font (default)

- **Site**: as below, change **Title**, **1x1 Widget Size**, **Group spacing**



Title

Node-RED IoT Dashboard

Options

Show the title bar

No swipe between tabs

Set theme in ui_template

Date Format

DD/MM/YYYY

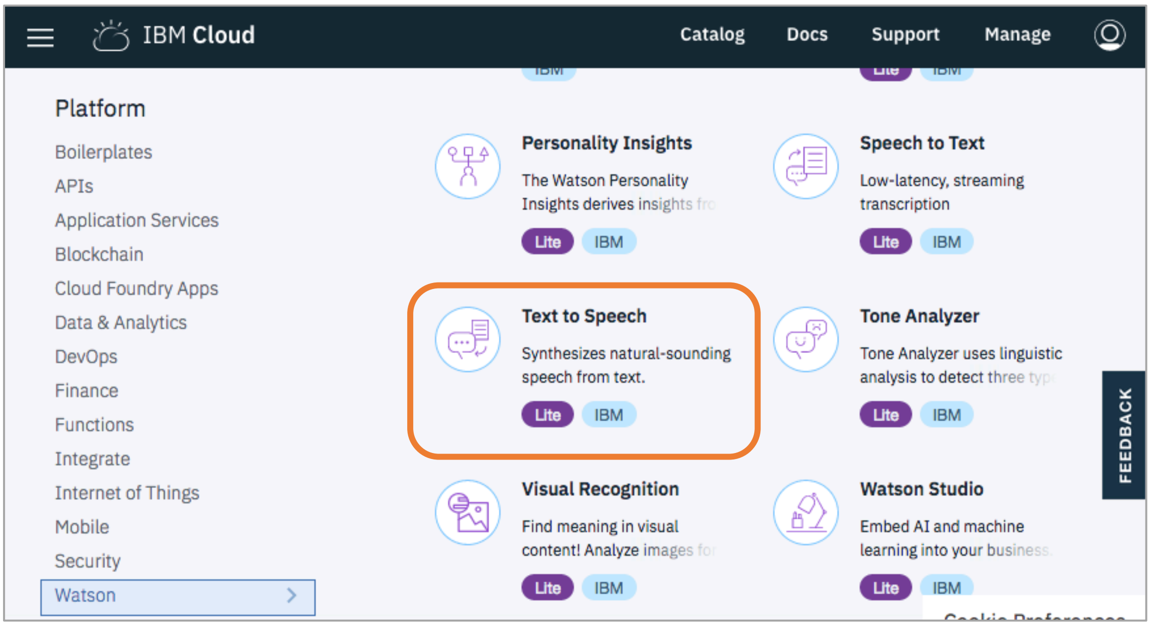
Sizes	Horizontal	Vertical
1x1 Widget Size	40	48
Widget Spacing	6	6
Group Padding	0	0
Group Spacing	5	6

__ 7. Deploy and check your dashboard.

Section 3. Add voice alert on dashboard

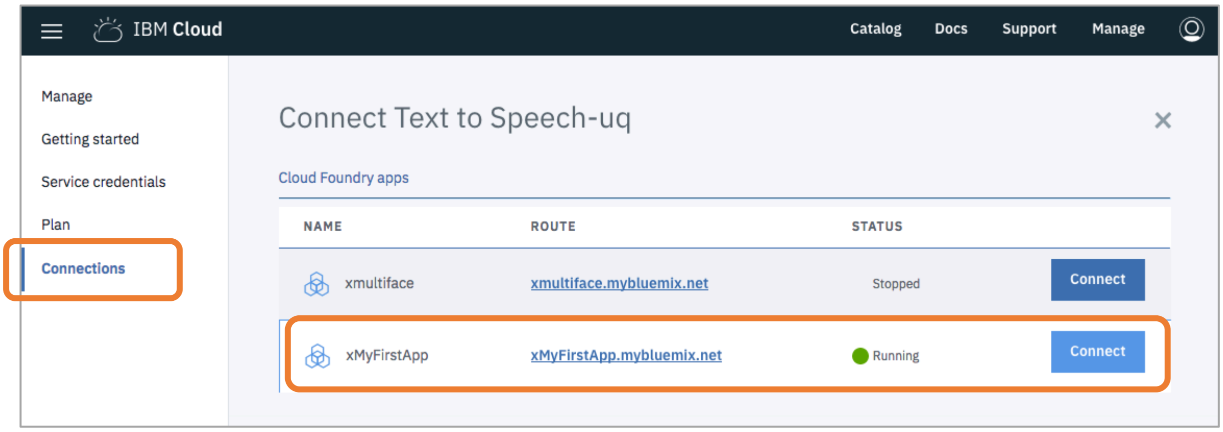
In this section, you will add a voice node allowing your app to tell say message when temperature change. To do that, you will deploy a new Watson service: Text to Speech.

__ 1. In the IBM Cloud window, click **Catalog** then **Watson** category.
Click on **Text to Speech** service to create your own instance.



__ 2. Enter a name for your service and click **Create**.

__ 3. On new service dashboard, click **Connection**, and connect Text to speech service to your Node-RED application: **Connect**.

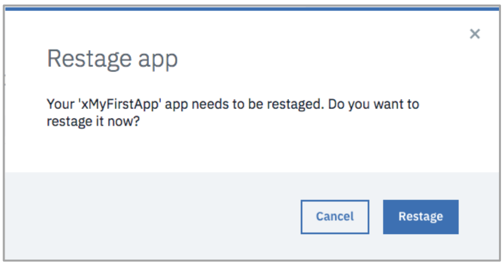


Connect Text to Speech-uq

Cloud Foundry apps

NAME	ROUTE	STATUS	
xmultiface	xmultiface.mybluemix.net	Stopped	Connect
xMyFirstApp	xMyFirstApp.mybluemix.net	● Running	Connect

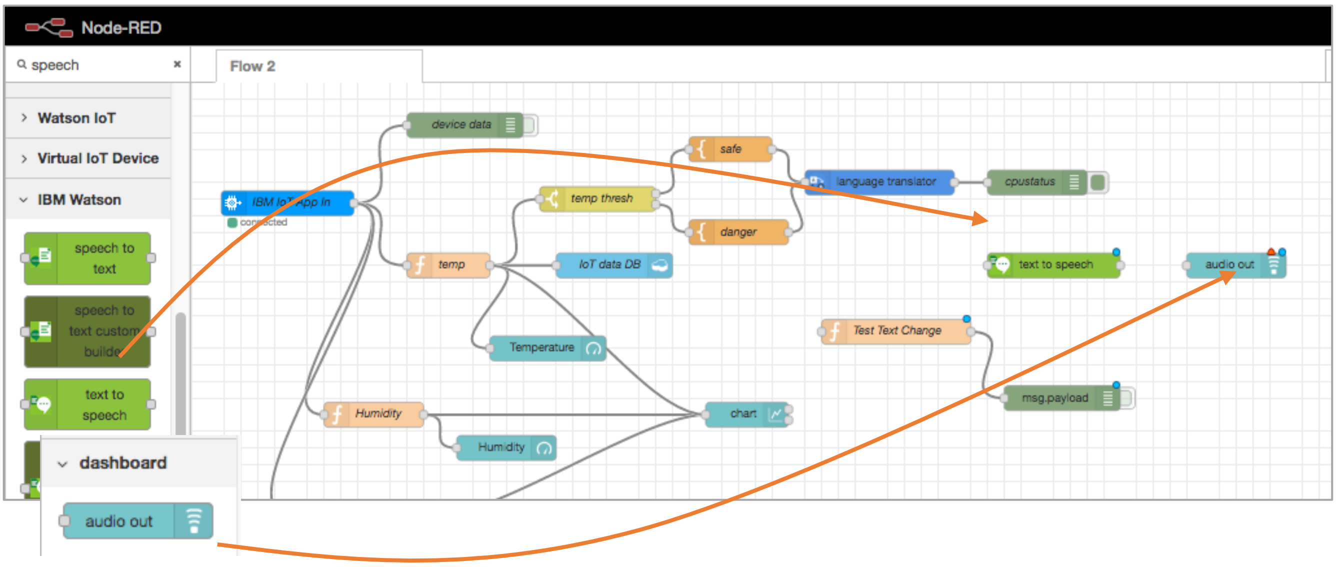
__ 4. Accept to restage application and wait for your Node-RED application to restart.



Restage app

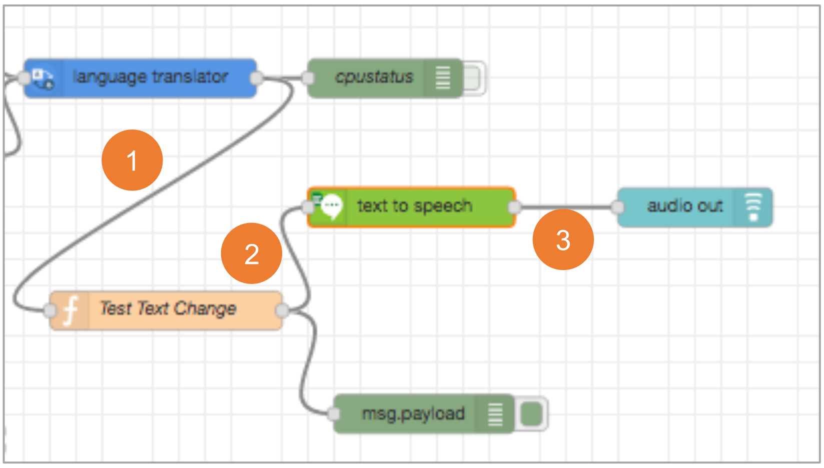
Your 'xMyFirstApp' app needs to be restaged. Do you want to restage it now?

__ 5. When restarted, in Node-RED environment, add an **Audio out** node (category **Dashboard**) and a **text to speech** node (category **Watson**) to your flow

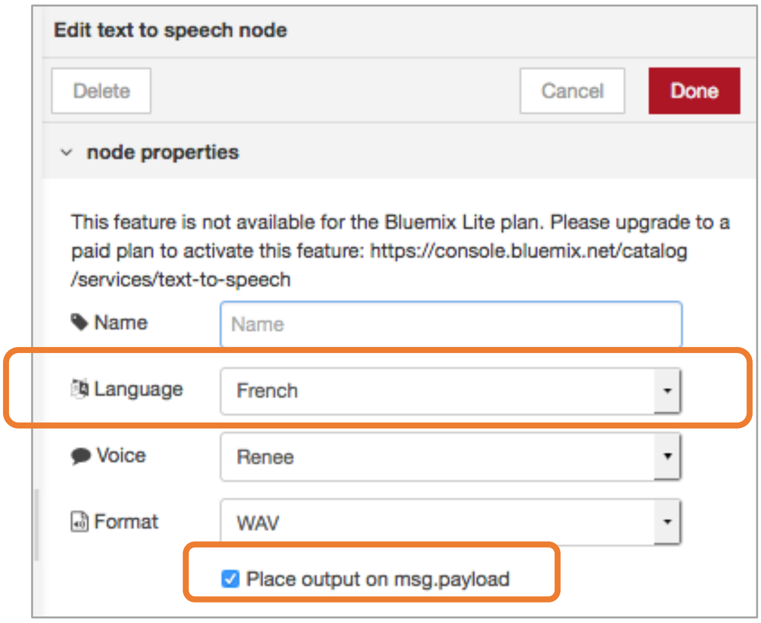


__ 8. Connect nodes as below

1. **Test text change** to **language translator**
2. **Test text change** to **text to speech**
3. **text to speech** to **audio out**



__ 9. Configure **text to speech** node to
- use the language your translated to (as configured before in **language translator** node)
- place the output on msg.payload (check box)



Edit text to speech node

This feature is not available for the Bluemix Lite plan. Please upgrade to a paid plan to activate this feature: https://console.bluemix.net/catalog/services/text-to-speech

Name: Name

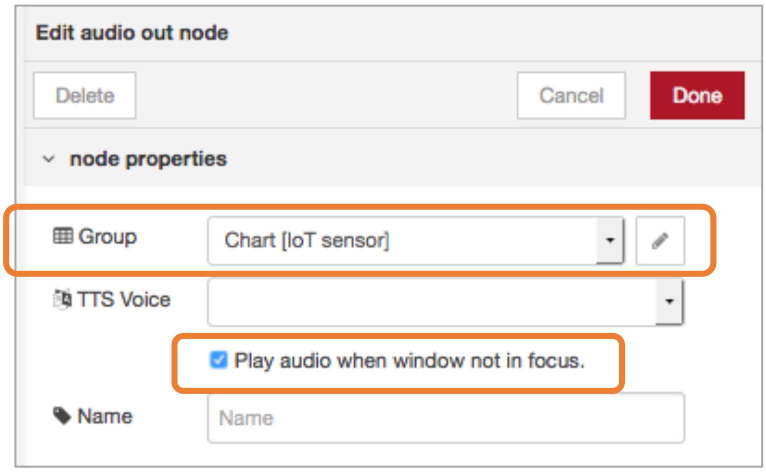
Language: French

Voice: Renee

Format: WAV

☑ Place output on msg.payload

__ 10. Configure **audio out** node
- Set **group** to **chart [IoT sensor]**
- Check **Play audio when window not in focus**



Edit audio out node

Group: Chart [IoT sensor]

TTS Voice:

☑ Play audio when window not in focus.

Name: Name

__ 11. Deploy the flow: **Deploy**

__ 12. Do you hear something ?
Try to change temperature in virtual sensor app.

______________________________ END OF LAB ______________________________