

# ST-DiscoveryKit-WatsonIoT-Workshop

---

STMicroelectronics Discovery Kit IoT Node and IBM Watson Hands-On Workshop

## ST Discovery Kit IoT Node and IBM Watson Hands-On Workshop:

---

Learn how to connect your next IoT design to the Cloud using IBM Watson IoT Platform and the ST Discovery Kit IoT node. The IoT Discovery Kit features an ultra-low-power STM32L4 Cortex®-M4 MCU, wireless connectivity and an array of sensors to showcase cloud connectivity features. Presenters: STMicroelectronics team with John Walicki, IBM

This hands-on workshop will show you how to integrate sensors, wireless connectivity (WiFi, NFC, BLE, SubGHz), a low-power microcontroller and sensor libraries into your next IoT design. You'll then learn to connect the Discovery Kit to IBM Cloud and Watson IoT Platform to create a new application in minutes using Node-RED.

The hands-on training is a working session – please bring your laptop. Note: Administrator rights is needed for software and driver installation. ST will provide the required STM32 development boards and software. SPACE IS LIMITED FOR THIS SESSION – FIRST COME, FIRST SEATED. Must be present and stay for training to receive free kit.

## Learning Objectives:

---

In this workshop, you will learn how to:

- Learn about the [STM32L4 Discovery Kit IoT node, low-power wireless, BLE, NFC, SubGHz, Wi-Fi](#) development board.
- Send Discovery Kit data to [Watson IoT Quickstart](#)
- Create an IoT Starter Kit application running in IBM Cloud
- Launch the Watson IoT Starter application
- Open the Watson IoT Platform so that you can send/receive data from the Discovery IoT device
- Create a Discovery IoT device type and device
- Configure the Node-RED visual programming editor
- Secure your Node-RED Editor in IBM Cloud
- Install additional Node-RED nodes
- Import a prebuilt flow from GitHub
- Create a new Node-RED flow and configure IoT Nodes

- Output the ST Microelectronics Discovery Kit IoT node temperature and humidity data
- Work with JSON data and observe the sensor results in the Debug sidebar
- Create a Node-RED Dashboard
- Experiment with Chart types
- Plot Real Time sensor data
- Trigger alerts when the real-time sensor data exceeds a threshold value
- Create a Node-RED flow that uses the Cloudant node
- Format a time series database record
- View Cloudant databases
- Read datasets from a Cloudant database
- Create a chart of historical data
- Set up a new Watson Studio Project
- Create a Jupyter Notebook
- Read data from a Cloudant DB into Spark
- Calculate the Average, Standard Deviation, and Find the Min/Max

## Prerequisites

---

This tutorial can be completed using an IBM Cloud Lite account.

- Create an [IBM Cloud account](#)
- Log into [IBM Cloud](#)

## Part 1 - ST Micro Discovery Kit IoT Node and Watson IoT Quickstart

---

- Instructions : Review Slides 5 to 13

## Part 2 - Create an Internet of Things Platform Starter application

---

- Instructions : [Create an Internet of Things Platform Starter application](#)

## Part 3 - Create a Watson IoT Platform Device Type and Device

---

- Instructions : [Create a Watson IoT Platform Device Type and Device](#)

## **Part 4 - Node-RED Set up and Configuration in IBM Cloud**

---

- Instructions : [Node-RED Set up and Configuration in IBM Cloud](#)

## **Part 5 - Receive Discovery Kit Environmental Sensor Data in Node-RED**

---

- Instructions : [Receive Discovery Kit Environmental Sensor Data in Node-RED](#)

## **Part 6 - Node-RED Dashboard Charts - Plot Environmental Sensor Data**

---

- Instructions : [Node-RED Dashboard Charts - Plot Environmental Sensor Data](#)

## **Part 7 - Store Data in Cloud Storage for Historical Data Analytics**

---

- Instructions : [Store Data in Cloud Storage for Historical Data Analytics](#)

## **Part 8 - Node-RED Charts of Historical Sensor Data**

---

- Instructions : [Node-RED Charts of Historical Sensor Data](#)

## **Part 9 - Watson Studio Set up and Configuration in IBM Cloud**

---

- Instructions : [Watson Studio Set up and Configuration in IBM Cloud](#)

## **Part 10 - Discovery Insights via Jupyter Notebook in Watson Studio**

---

- Instructions : [Watson Studio Jupyter Notebook](#)

## References:

---

- Download link to [TrueStudio](#)
- Download link to the [IBM Watson IoT examples](#) that are buildable in TrueStudio

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Create an Internet of Things Platform Starter application

---

In this tutorial you will learn how to create a Watson IoT Platform Starter application that can be used to connect IoT devices and simulated IoT devices to the Watson IoT Platform. The **Internet of Things Platform Starter** is designed with pre-assembled services that work together. The Internet of Things Platform Starter includes a Node-RED Node.js web server, Cloudant database to store the sensor data, and the IoT platform service so you can connect devices.

## Learning objectives

---

After completing this tutorial you will be able to:

- Create an IoT Starter Kit application running in IBM Cloud
- Launch the Watson IoT Starter application
- Open the Watson IoT Platform so that you can send data from IoT devices and device simulators

## Prerequisites

---

This tutorial can be completed using an IBM Cloud Lite account.

- Create an [IBM Cloud account](#)
- Log into [IBM Cloud](#)

## Estimated time

---

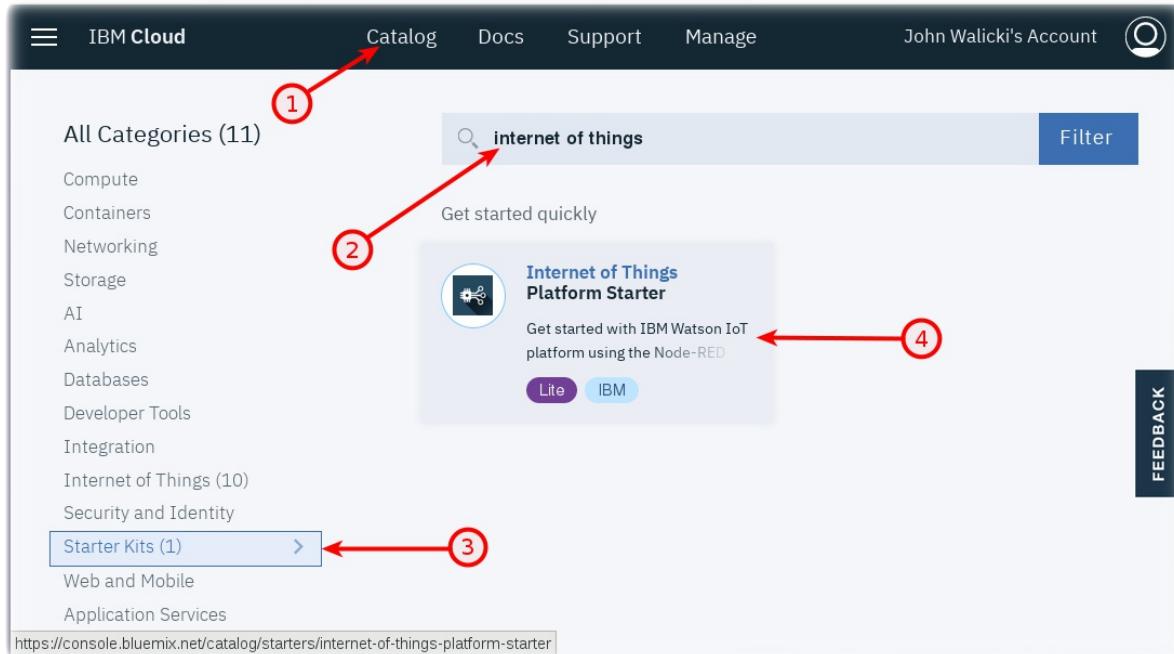
You can complete this task in no more than 20 minutes.

## Step 1 - Create a Watson IoT Starter Application

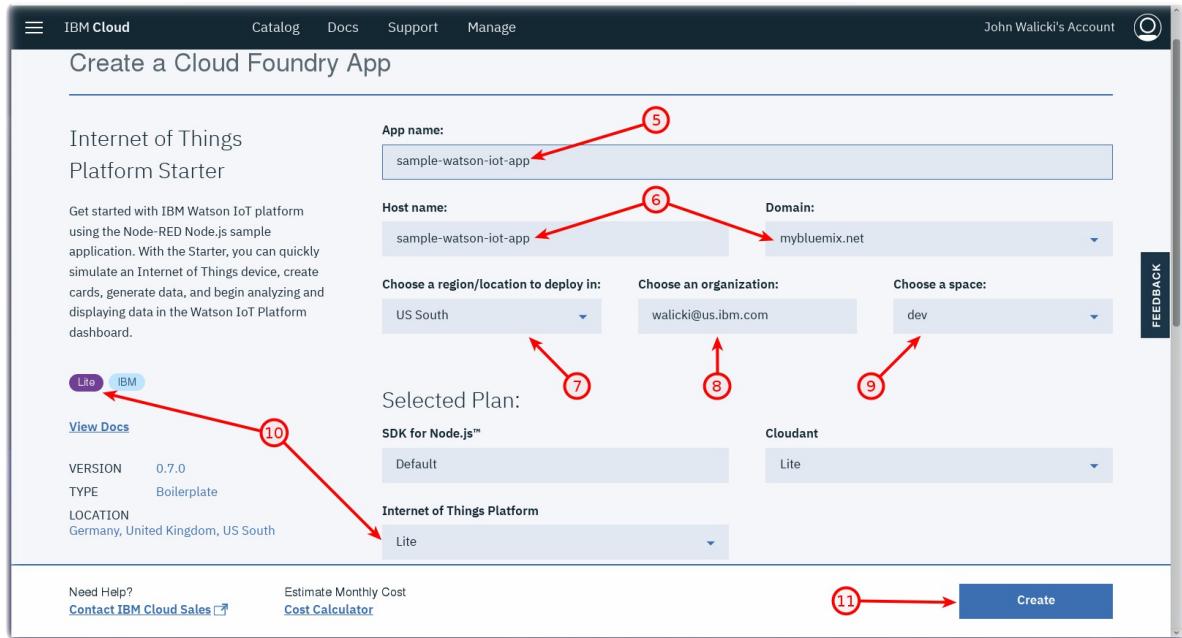
---

Follow these steps to create a Watson IoT Platform Starter application in IBM Cloud.

- Create an [IBM Cloud account](#) and log into [IBM Cloud](#)
- Click on the [Catalog](#) (1) and search for **internet of things** (2)
- Under **Starter Kits** (3) click on **Internet of Things Platform Starter** (4)



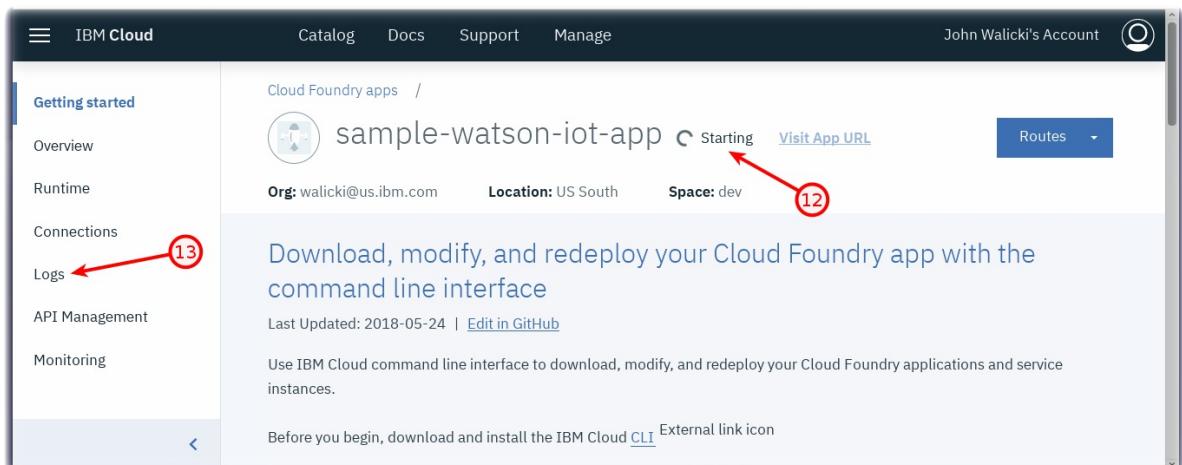
- Enter a **unique name** for your application (5). For example, *stmicro-discoverykit-yourname*. This name will be part of the **application URL** (6)
  - Note: In case the name is not unique you will receive an error message and can enter another name.
- The **Region** (7), **Organization** (8) and **Space** (9) will be prepopulated with valid options for your IBM Cloud account. If you have a Lite account then accept the defaults. If you have a trial or paid account, or belong to additional organizations, then you may choose to deploy in to any region, organization and space available to you.
- The IoT Starter application can be provisioned in the IBM Cloud *Lite* plan (10)
- Click on the **Create** button (11)



## Step 2 - Launch the IoT Starter Application

The Internet of Things Starter application will be provisioned in the IBM Cloud region that was specified. This is called staging an application. It can take a few minutes for this process to complete.

- Wait for the application to provision and start (12)
  - Note: While you wait, you can click on the Logs tab and review the activity logs from the platform and Node.js runtime.
  - Click on **Logs** (13) in the left navigation menu.
  - Scroll to the bottom to check the latest DevOps messages.



## Step 3 - Launch the Watson IoT platform

Return to the IBM Cloud Application Details page.

- Click on the **Overview** item (14) in the left navigation menu.
- Click on your *iotf-service* (15) in the **Connections** tile.

The screenshot shows the IBM Cloud Application Details page for the app 'stmicro-discovery-kit-iot-node'. The left sidebar has 'Overview' selected (circled in red, labeled '14'). The main area shows runtime details: Buildpack (noruntime), Instances (1), MB Memory per Instance (256), and Total MB Allocation (256). Below this is a 'Connections (2)' section where 'iotf-service' is listed (circled in red, labeled '15').

- The **Internet of Things** service landing page will open. Click on the **Launch** (16) button.

The screenshot shows the Watson IoT Platform service landing page for the app 'stmicro-discovery-kit-iot-node-iotf-service'. It features a 'Let's get started with Watson IoT Platform' section with a subtext about securely connecting devices. The 'Launch' button at the bottom is circled in red and labeled '16'.

- The **Watson IoT Platform** service will open in a new browser tab and you can create an IoT device.

IBM Watson IoT Platform

walicki@us.ibm.com  
ID: tspm30

Browse Action Device Types + Add Device

## Browse Devices

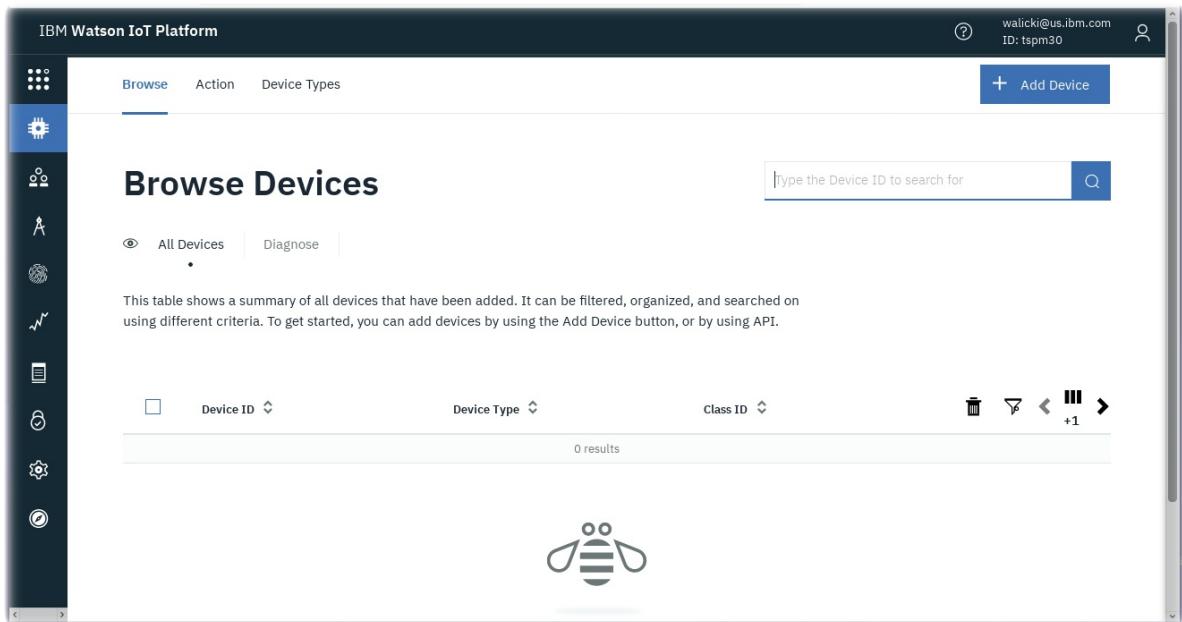
Type the Device ID to search for  Q

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

<input type="checkbox"/>	Device ID	Device Type	Class ID	<span>trash</span>	<span>down arrow</span>	<span>left arrow</span>	<span>right arrow</span>	+1
0 results								





**Congratulations** - You have completed the creation of a Watson Internet of Things Platform Starter application in IBM Cloud.

Continue to the next step - [Device Types and Devices](#)

---

*Quick links :*

[Home - IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

---

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Registering a new device to the Watson IoT Platform

## Lab Objectives

This Lab will show you how to register your ST Microelectronics Discovery Kit IoT Node with the IBM Watson Internet of Things Platform. In the lab you will learn:

- How to define a device type and register a device in the Watson IoT Platform

## Introduction

Before you can connect a device to the Watson IoT Platform you need to define how the device will connect to the platform and also register the device to generate an access token for the device. This will be used to verify the device identity.

You need to decide how you want to group devices, by function, by hardware type, etc. Each device registered on the platform must be registered against a device type. There are no restrictions about how devices are grouped and the device types, for this workshop we will create a device type representing the ST Microelectronics Discovery Kit IoT Node devices.

## Step 1 - Add a new device type for Discovery Kit IoT Node device

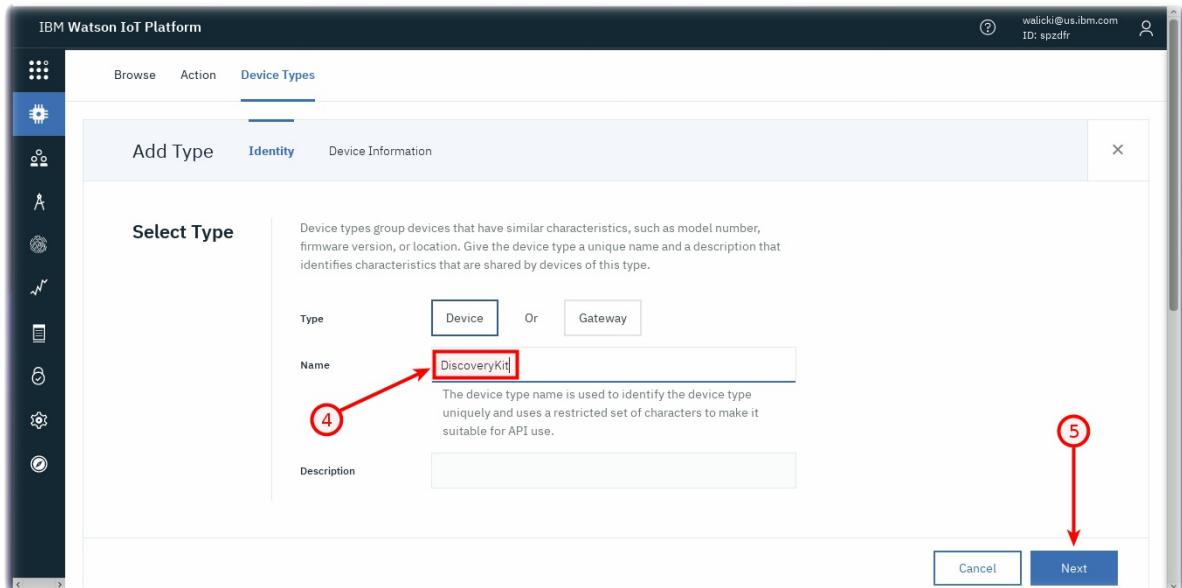
- Navigate into the **Devices** section (1) of the console

The screenshot shows the 'Browse Devices' section of the IBM Watson IoT Platform. On the left is a vertical toolbar with various icons. At the top, there are tabs for 'Browse', 'Action', and 'Device Types'. A red circle labeled '1' points to the 'Device Types' tab, which is currently selected. To the right of the tabs is a search bar with placeholder text 'Type the Device ID to search for' and a magnifying glass icon. Below the search bar is a table header with columns: 'Device ID', 'Device Type', 'Class ID', and 'Date Added'. Underneath the table, it says '0 results'. In the center, there's a small bee icon and the text 'You don't have any devices.'.

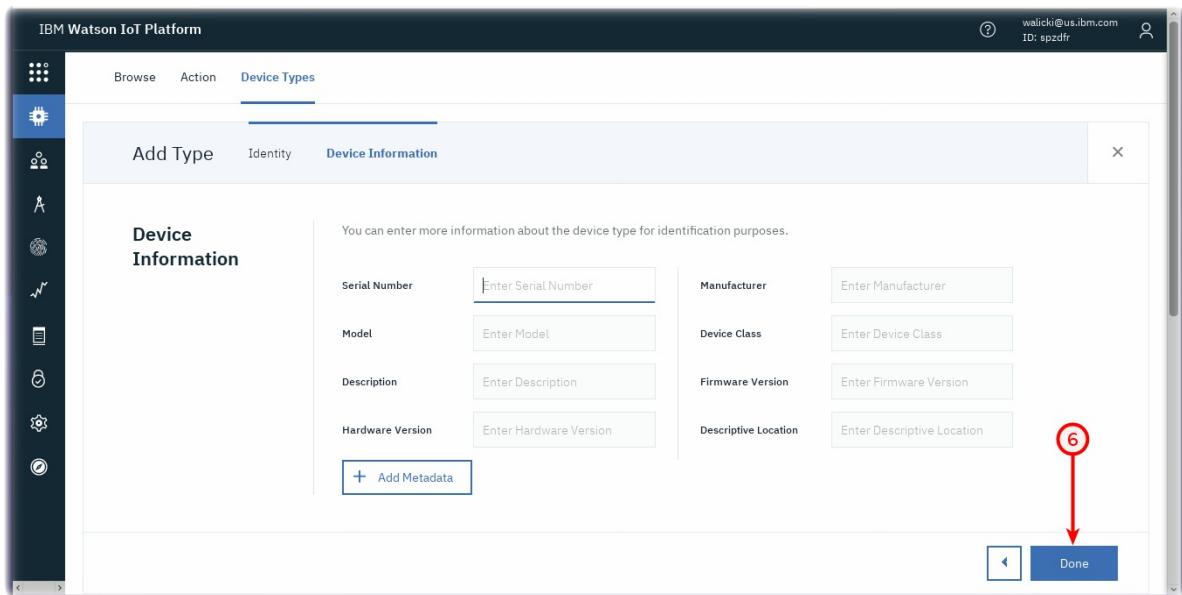
- Select the **Device Types** section (2). Press the **+ Add Device Type** (3) button

The screenshot shows the 'Device Types' section of the IBM Watson Platform. The interface is similar to the 'Browse Devices' page, with a vertical toolbar on the left, a 'Device Types' tab highlighted by a red box and labeled '2', and a 'Add Device Type' button at the top right labeled '3'. A search bar is present at the top right. The main area displays a table with columns: 'Name', 'Description', 'Number of Devices', and 'Class ID'. Below the table, it says 'You don't have any device types created.' and features a 'Add Device Type' button.

- Enter the following:
  - Type : Ensure **Device** is selected (NOT Gateway)
  - Name : Enter **DiscoveryKit** (4)
- Press the **Next** button (5)



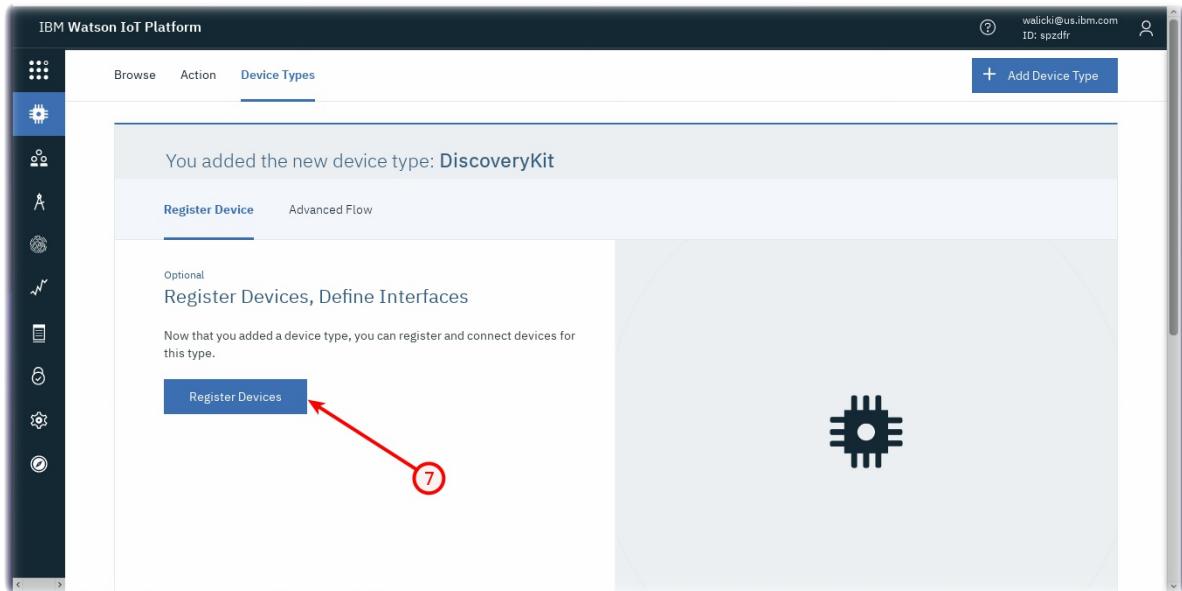
- Press the **Done** button (6)



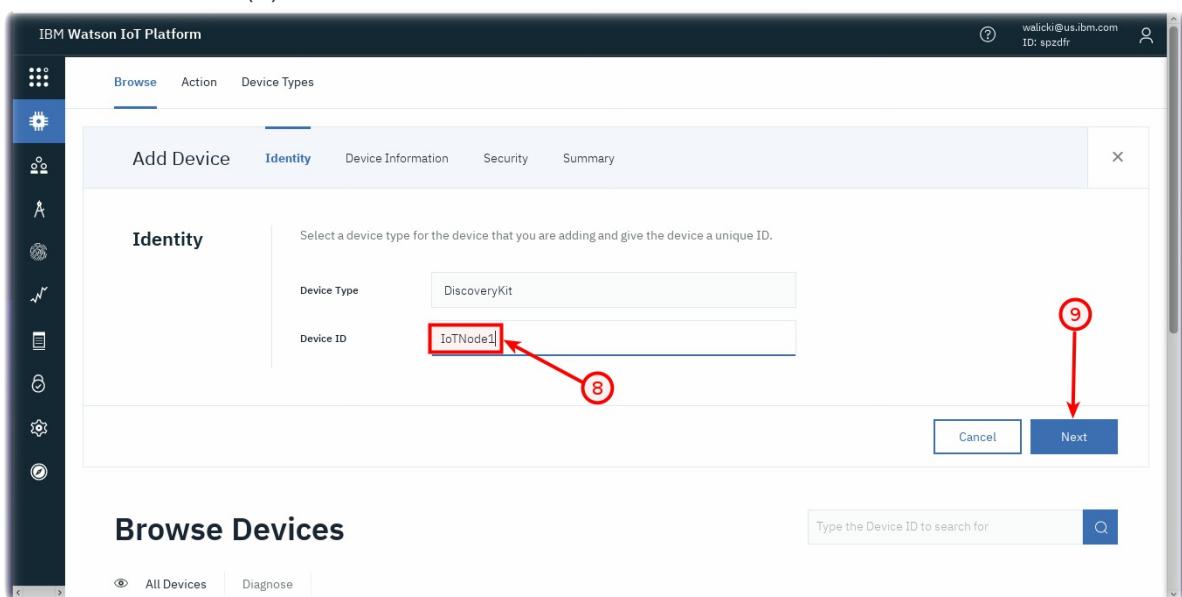
## Step 2 - Register a new DiscoveryKit device in the IoT Platform

You now have the opportunity to register a device.

- Press **Register Devices** (7)



- The DiscoveryKit device type should be pre-selected. You now need to enter a unique device ID. You can choose how you want to identify devices. For the workshop, use a simple format, such as **IoTNode1**. (8)
- Press **Next** button (9)



- Press **Next** button (10)

IBM Watson IoT Platform

walicki@us.ibm.com  
ID: spzdrf

Browse Action Device Types

Add Device Identity Device Information Security Summary X

**Device Information**

You can modify the default device information and enter more information about the device for identification purposes.

Serial Number	<input type="text" value="Enter Serial Number"/>	Manufacturer	<input type="text" value="Enter Manufacturer"/>
Model	<input type="text" value="Enter Model"/>	Device Class	<input type="text" value="Enter Device Class"/>
Description	<input type="text" value="Enter Description"/>	Firmware Version	<input type="text" value="Enter Firmware Version"/>
Hardware Version	<input type="text" value="Enter Hardware Version"/>	Descriptive Location	<input type="text" value="Enter Descriptive Location"/>

+ Add Metadata

10

Next

- You will be prompted to provide a token (11)
- Each time you connect the device the token will need to be presented to the server. Once the device is registered there is no way to recover a token. You will need to delete and reregister the device if the token is lost or forgotten.
- Enter a **token** for your device (11) then press **Next** (12).

IBM Watson IoT Platform

walicki@us.ibm.com  
ID: spzdrf

Browse Action Device Types

Add Device Identity Device Information Security Summary X

**Device Security**

There are two options for selecting a device authentication token.

**Auto-generated authentication token (default)**

Allow the service to generate an authentication token for you. Tokens are 18 characters and contain a mix of alphanumeric characters and symbols. The token is returned to you at the end of the device registration process.

**Self-provided authentication token**

Provide your own authentication token for this device. The token must be between 8 and 36 characters and contain a mix lowercase and uppercase letters, numbers, and symbols, which can include hyphens, underscores, and periods. Do not use repeated characters, dictionary words, user names, or other predefined sequences.

Authentication Token  11

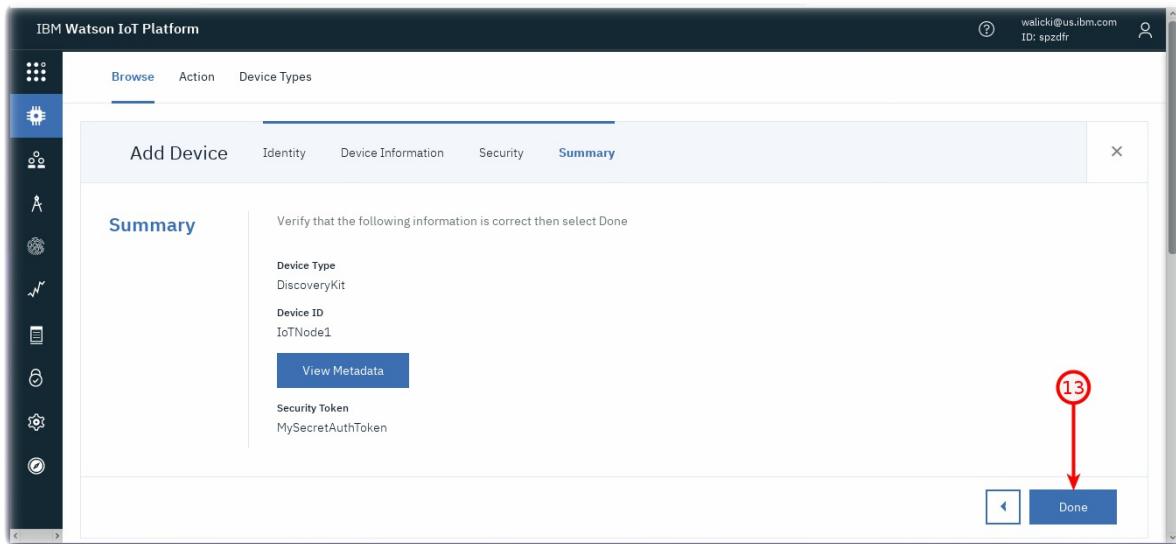
Make a note of the generated token. Lost authentication tokens cannot be recovered. Tokens are encrypted before being stored.

Authentication token are encrypted before we store them.

12

Next

- You will see a summary of the device. Press **Done** to complete the device registration.



- You are now shown a **Device Credentials** page - this is the last chance you get to see the token. Once you leave this page the token can not be recovered. Write down the Org, Device Type, Device ID and Authentication Token. You might even consider taking a screen shot.

**Device Credentials**

You registered your device to the organization. Add these credentials to the device to connect it to the platform. After the device is connected, you can navigate to view connection and event details.

Organization ID	spzdfr
Device Type	DiscoveryKit
Device ID	IoTNode1
Authentication Method	use-token-auth
Authentication Token	MySecretAuthToken

**⚠️ Authentication tokens are non-recoverable. If you misplace this token, you will need to re-register the device to generate a new authentication token.**

Find out how to add these credentials to your device [»](#)

## Step 3 - Reprogram the ST Microelectronics Discovery Kit IoT Node

Instead of sending the ST Microelectronics Discovery Kit IoT Node sensor data to Quickstart, we can now reprogram the board to send data to your secure Watson IoT Platform instance. If you have not yet powered the ST Microelectronics Discovery Kit IoT Node, connect it to your laptop using a MicroUSB cable. Start the terminal program to access the program running on the ST Microelectronics Discovery Kit IoT Node.

- Press the **Black Reset** on the ST Microelectronics Discovery Kit IoT Node
- Let it reconnect to the WiFi network
- When the program prompts, Enter **y** to update the device credentials

```
Do you want to update your way to register, or the device credentials? (y/n)
```

- When the program prompts, Enter **2** to specify a simple registered device.

```
Enter Registration Mode (1 - Quickstart, 2 - Simple):
```

- The program then prompts for your Org, DeviceType, Deviceld and Auth Token. You might want to type the string into a text editor to avoid typos and mistakes, then copy / paste the string into the program as hidden text. In the example case:

```
OrgId=spzdf;r;DeviceType=DiscoveryKit;DeviceId=IoTNode1;Token=MySecret
```

- The program will connect via MQTT to Watson IoT Platform.
- To send data, press the Blue **user** button. The application publishes the current sensor values.
- On double button push, the application enters in a loop and publishes sensor values automatically every second.
- For comparison, a full execution of the program is included below:

```
*****
*** STM32 IoT Discovery kit for STM32L475/STM32F413/STM32F769 MCU ***
*** X-CUBE-WATSON-X Cloud Connectivity Demonstration ***
*** FW version 1.0.1 - Jul 26 2018, 11:10:45 ***
*****  
  

*** Board personalization ***  
  

*** WIFI connection ***  
  

Push the User button (Blue) within the next 5 seconds if you want to update the WiFi netwo  
  

Initializing the WiFi module
Module initialized successfully: Inventek eS-WiFi ISM43362-M3G-L44-SPI C3.5.2.3.BETA9
Retrieving the WiFi module MAC address: c4:7f:51:03:f1:a9  
  

Connecting to AP: stmiot32 Attempt 1/3 ...
Connected to AP stmiot32
```

```

Mac address: c4:7f:51:03:f1:a9
Retrieving the IP address.
IP address: 192.168.1.236
Do you want to update your way to register, or the device credentials? (y/n)

Enter Registration Mode (1 - Quickstart, 2 - Simple):
You have selected the Simple registration mode.

Enter the Bluemix connection string of your device: (template: OrgId=xxx;DeviceType=xxx;DeviceConnection
connection string: --->
SimpleReg;OrgId=spzdfr;DeviceType=DiscoveryKit;DeviceId=IoTNode1;Token=MySecret
<---
Setting the RTC from the network time.
Configuring the RTC from Date: Tue, 04 Sep 2018 03:55:15 GMT
Device Client Connected to spzdfr.messaging.internetofthings.ibmcloud.com in registered mode

----- Callback following our request to become a managed device -----
Status :: 200
requestId : A27FD802-C29A8-4FA28-A0DAA-9F755DA3126B
Payload is : {"reqId":"A27FD802-C29A8-4FA28-A0DAA-9F755DA3126B", "rc":200}
-----

You can see your published data at https://spzdfr.internetofthings.ibmcloud.com
Sign in. Open a session with the "spzdfr" organization (next to your account name)
Select the device dashboard and click on the "IoTNode1" device to get its information and

When user button is pushed shortly, application publishes the sensor values, a 0/1 toggle
On double button push, application enters in a loop and publishes automatically every second

Enter the sensor values publication loop.
publishing sensor data
publishing sensor data
publishing sensor data

```

## Discovery Kit IoT Node Environmental Sensor data in Watson IoT platform

Confirm in Watson IoT Platform that ST Microelectronics Discovery Kit IoT Node environmental sensor data is arriving successfully.

IBM Watson IoT Platform

walicki@us.ibm.com  
ID: spzdrf

Browse Action Device Types

+ Add Device

Type the Device ID to search for

## Browse Devices

All Devices | Diagnose

This table shows a summary of all devices connected to the platform. You can filter devices using different criteria. To get started, click on a device to view its details.

Device ID	Identity	Device Information
IoTNode1		Showing Raw Data

**Event Payload**

```

1  {
2   "d": {
3     "temperature": 27.9,
4     "humidity": 47.6,
5     "pressure": 1009.95,
6     "proximity": 8190,
7     "acc_x": -18,
8     "acc_y": -4,
9     "acc_z": 1018,
10    "gyr_x": 350,
11    "gyr_y": -1050,
12    "gyr_z": 840,
13    "mag_x": -51,
14    "mag_y": -354,
15    "mag_z": 399
16  }
17 }
```

**Event**

status	{ "d": { "Led state": 1, "Timestamp": "2018-09-04T11:51:11Z" }}	json	2 minutes ago
status	{ "d": { "temperature": 27.9, "humidity": 47.6, "pressure": 1009.95, "proximity": 8190, "acc_x": -18, "acc_y": -4, "acc_z": 1018, "gyr_x": 350, "gyr_y": -1050, "gyr_z": 840, "mag_x": -51, "mag_y": -354, "mag_z": 399 } }	json	2 minutes ago
status	{ "d": { "Led state": 0, "Timestamp": "2018-09-04T11:51:11Z" }}	json	2 minutes ago
status	{ "d": { "temperature": 27.92, "humidity": 47.6, "pressure": 1009.95, "proximity": 8190, "acc_x": -18, "acc_y": -4, "acc_z": 1018, "gyr_x": 350, "gyr_y": -1050, "gyr_z": 840, "mag_x": -51, "mag_y": -354, "mag_z": 399 } }	json	2 minutes ago

**Congratulations :** Your board is sending data securely to Watson IoT Platform using TLS encrypted MQTT packets and server certificates.

Continue to the next step - [Node-RED Setup](#)

*Quick links :*

[Home - IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Node-RED Set up and Configuration in IBM Cloud

## Lab Objectives

In this lab you will set up Node-RED in your Watson IoT Starter application created at the end of Part 2. You will learn:

- Node-RED Visual Programming
- How to secure your Node-RED Editor in IBM Cloud
- How to install additional Node-RED nodes
- How to import a prebuilt flow from GitHub

## Introduction

Node-RED is an open-source Node.js application that provides a visual programming editor that makes it easy to wire together flows.

## Step 1 - Node-RED Visual Programming

Recall that in Part 2, you created a IoT Starter Application. Once the Green **Running** icon appears, click the **View App URL** button (6)

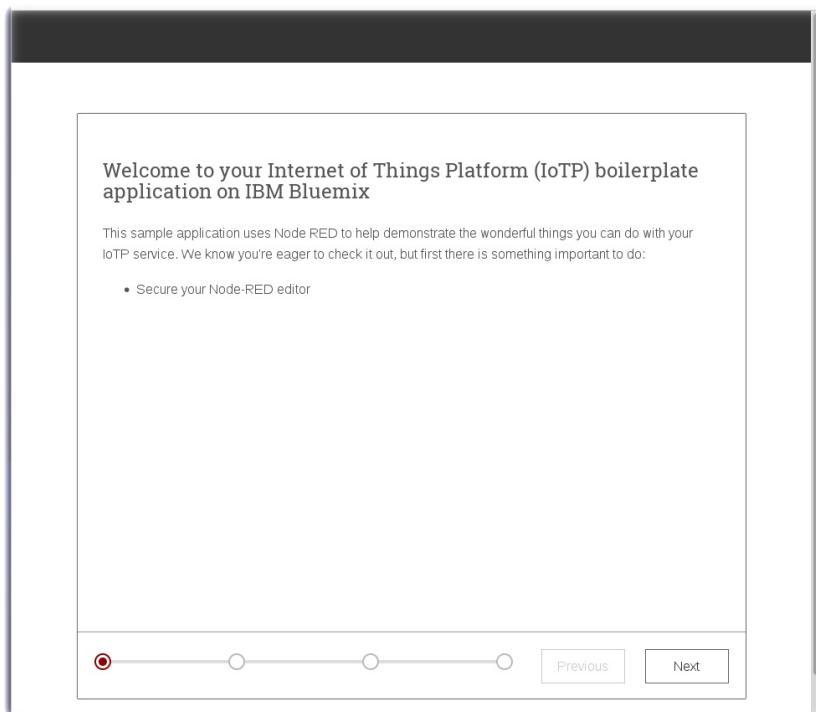
The screenshot shows the IBM Cloud dashboard with the 'Cloud Foundry apps' section selected. The application 'stmicro-discovery-kit-iot-node' is listed, showing it is running with 6 instances. A red arrow points to the 'Visit App URL' link. The sidebar on the left includes links for 'Getting started', 'Overview', 'Runtime', 'Connections', 'Logs', 'Monitoring', and 'API Management'.

## Step 2 - How to secure your Node-RED Editor in IBM Cloud

A new browser tab will open to the Node-RED start page. Node-RED is an open-source Node.js application that provides a visual programming editor that makes it easy to wire together flows.

Several panels will help you set up Node-RED in your Watson IoT Starter application.

- Welcome to Node-RED and IoT Platform. Click the **Next** button to proceed. Step 1 of 4



- Secure your Node-RED editor by setting a username / password. Remember your username / password. Click the **Next** button to proceed. Step 2 of 4

Secure your Node-RED editor

Secure your editor so only authorised users can access it

Username

Password  Must be at least 8 characters

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

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

Previous Next

- If you forget, you can reset the username / password in the Cloudant DB or by setting IBM Cloud environment variables. Click the **Finish** button to proceed. Step 3 of 4

Finish the configuration

You have made the following selections:

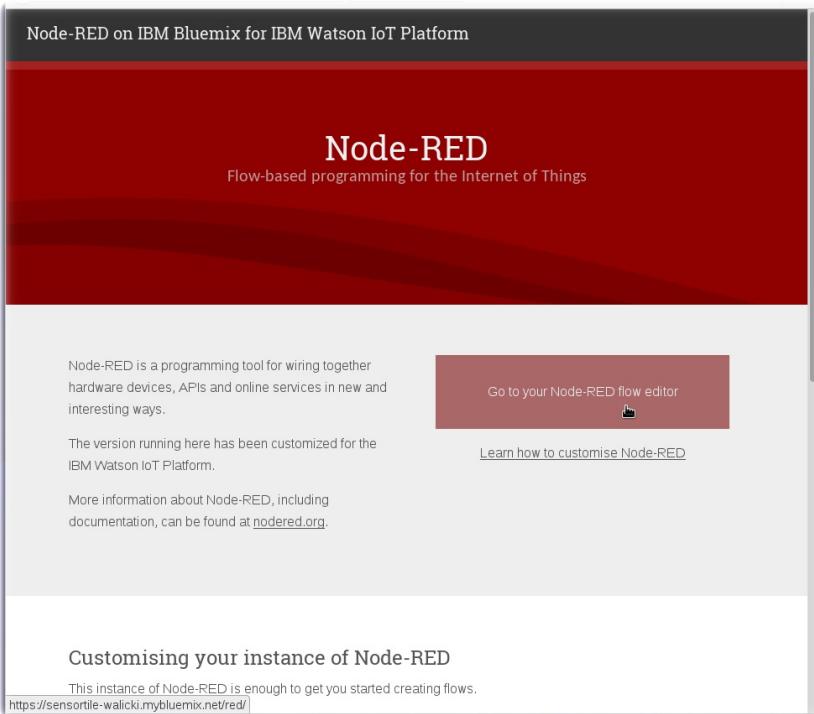
- Secure your editor so only authorised users can access it

The settings will be persisted in the CloudantDB bound to this application. You can override them at any time by setting the following environment variables via the Bluemix console:

- NODE\_RED\_USERNAME - the username
- NODE\_RED\_PASSWORD - the password
- NODE\_RED\_GUEST\_ACCESS - if set to 'true', allows anyone read-only access to the editor

Previous Finish

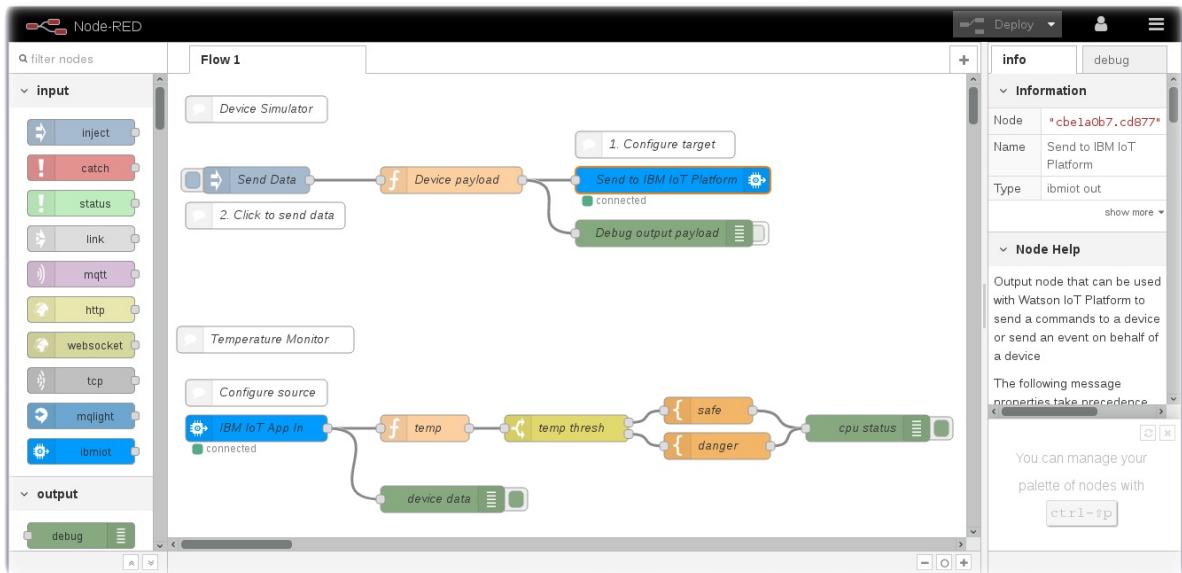
- Click the **Go to your Node-RED flow editor** button to launch the Node-RED flow editor.  
Step 4 of 4



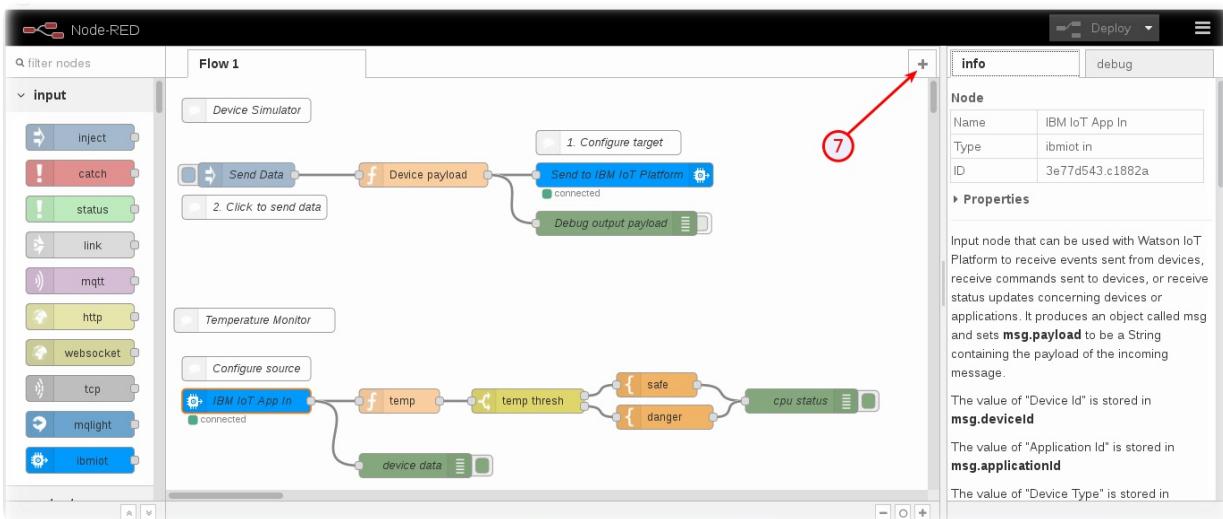
- **Sign in** with your new username and password credentials.



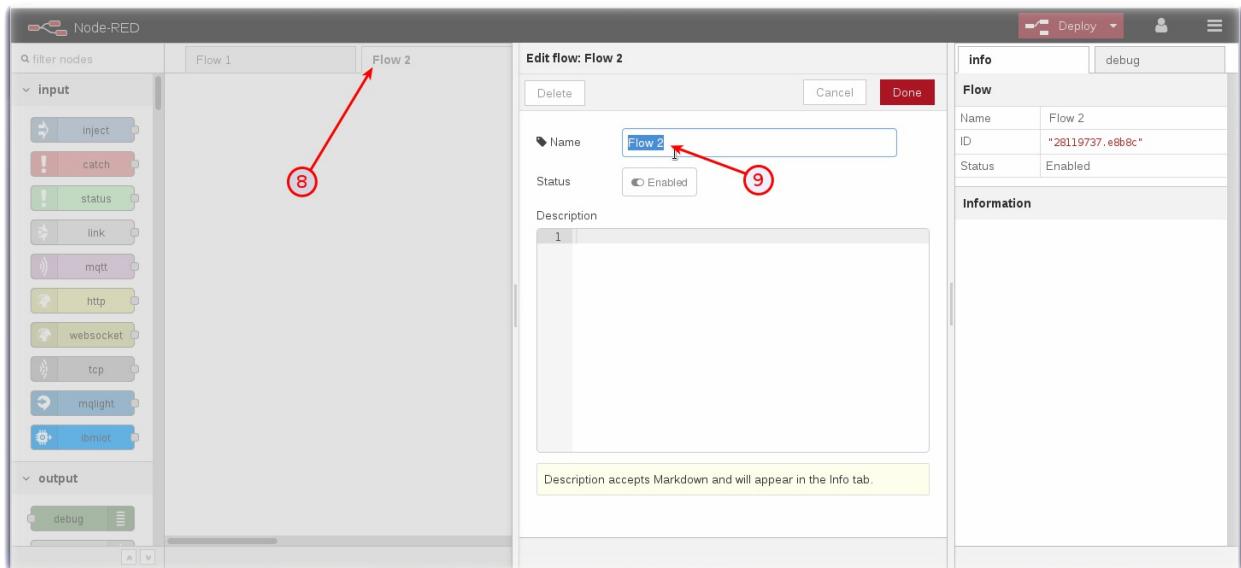
- The **Node-RED Visual Programming Editor** will open with a default flow. On the left side is a **palette of nodes** that you can drag onto the flow. You can **wire nodes together** to create a program.



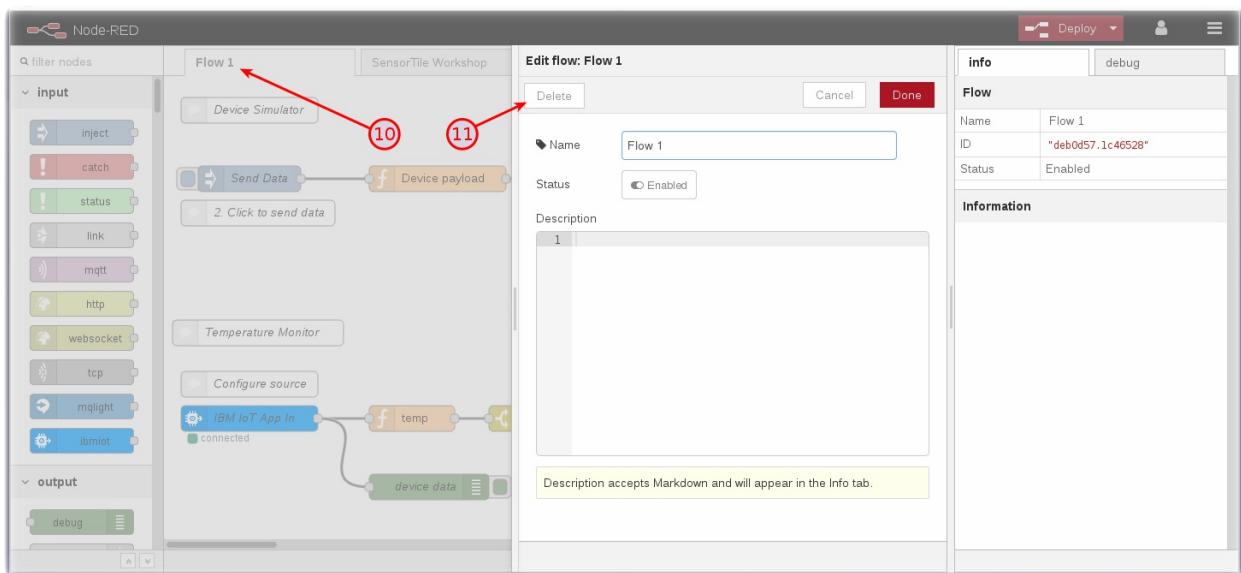
- The **Node-RED Visual Programming Editor** will open with a default flow.
- On the left side is a **palette of nodes** that you can drag onto the flow.
- You can **wire nodes together** to create a program.
- The sample IoT Starter flow is not applicable to this workshop and can be deleted.



- Click the + icon (7) to add a new tab. Double Click on the **Flow 2** tab header (8).
- Rename this tab from **Flow 2** to Receive Discovery Kit Data (9)

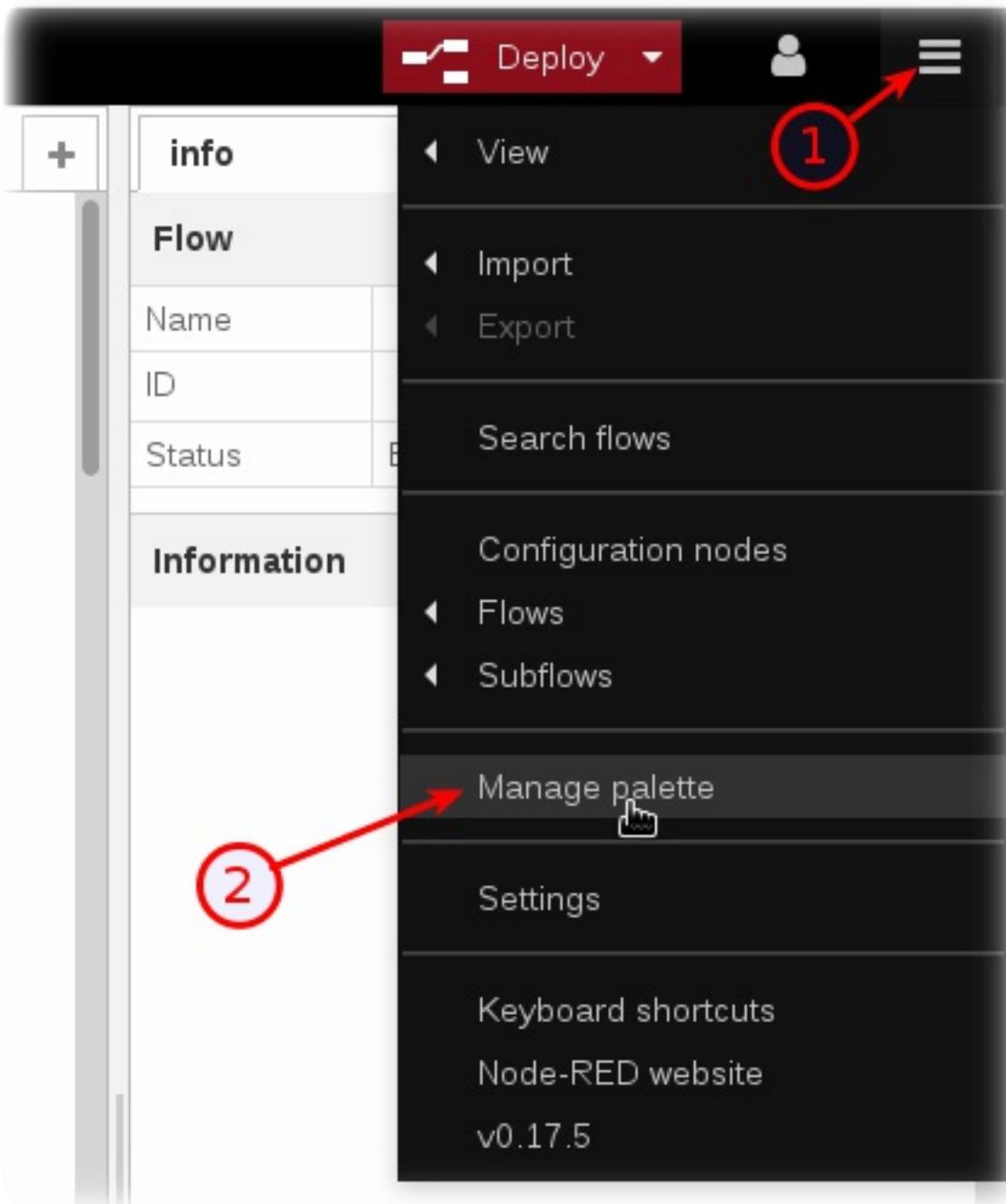


- Double Click on the **Flow 1** tab header (10). Press the **Delete** button. (11)

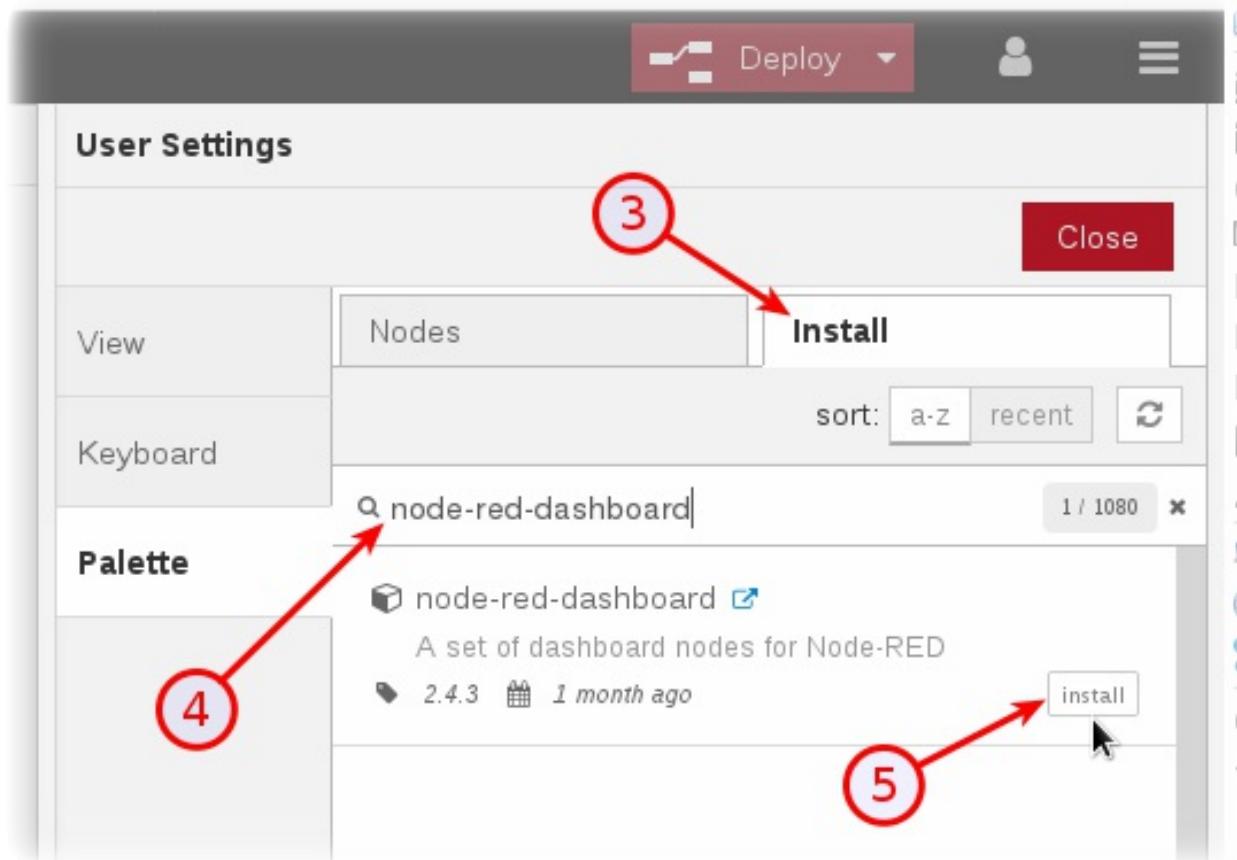


## Step 3 - How to install additional Node-RED nodes

- The IoT Starter Application deployed into IBM Cloud includes just a small subset of Node-RED nodes. The Node-RED palette can be extended with over one thousand additional nodes for different devices and functionality. These NPM nodes can be browsed at <http://flows.nodered.org>
- In this Step, you will add the **Node-RED Dashboard** nodes to your Internet of Things Starter Application.
- Click on the Node-RED **Menu** (1) in the upper right corner, then **Manage palette** (2)



- Turn to the **Install** tab (3), type **node-red-dashboard** (4) and press the **Install** button (5).

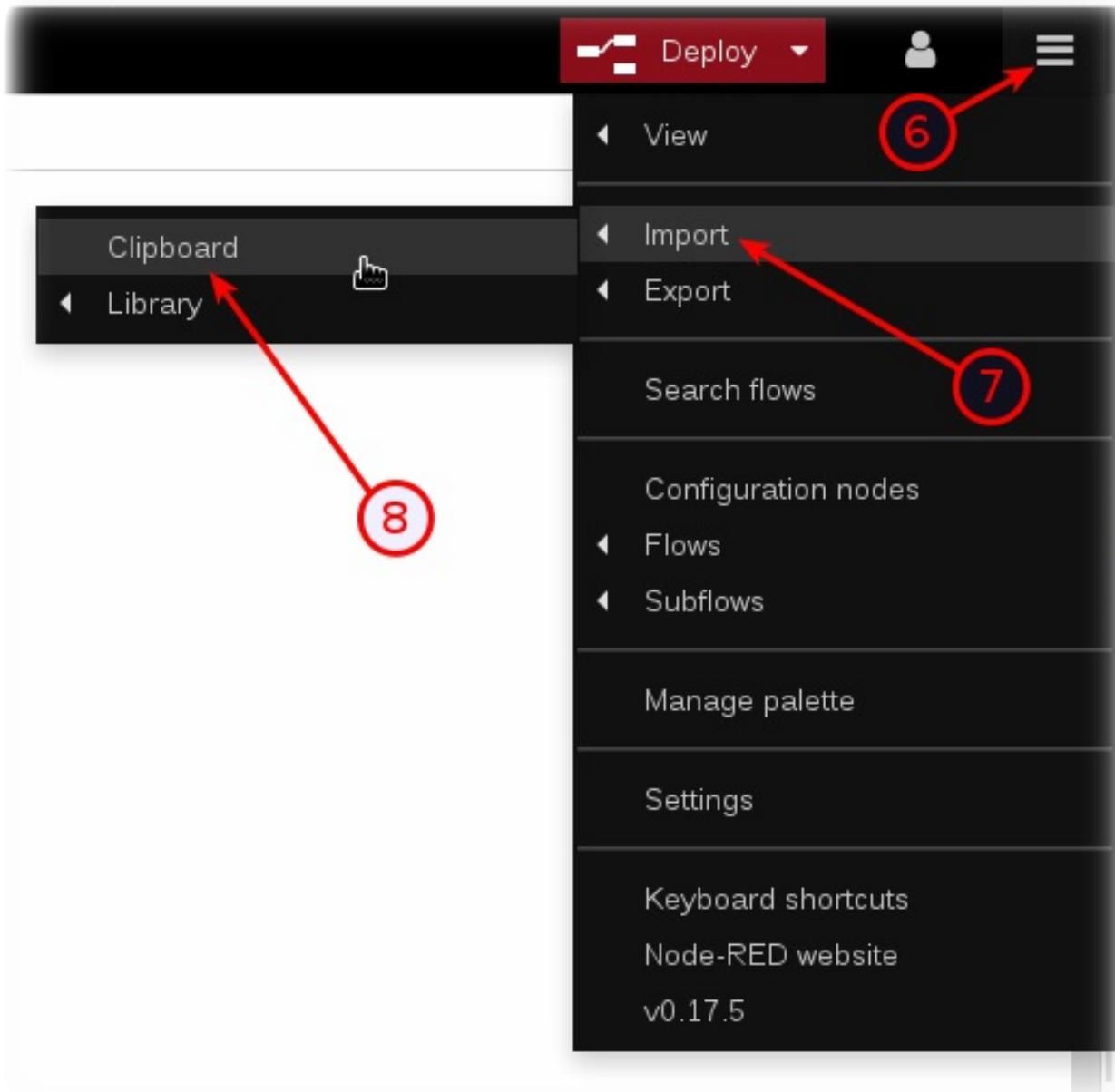


- Press the **Install** button in the next dialog.

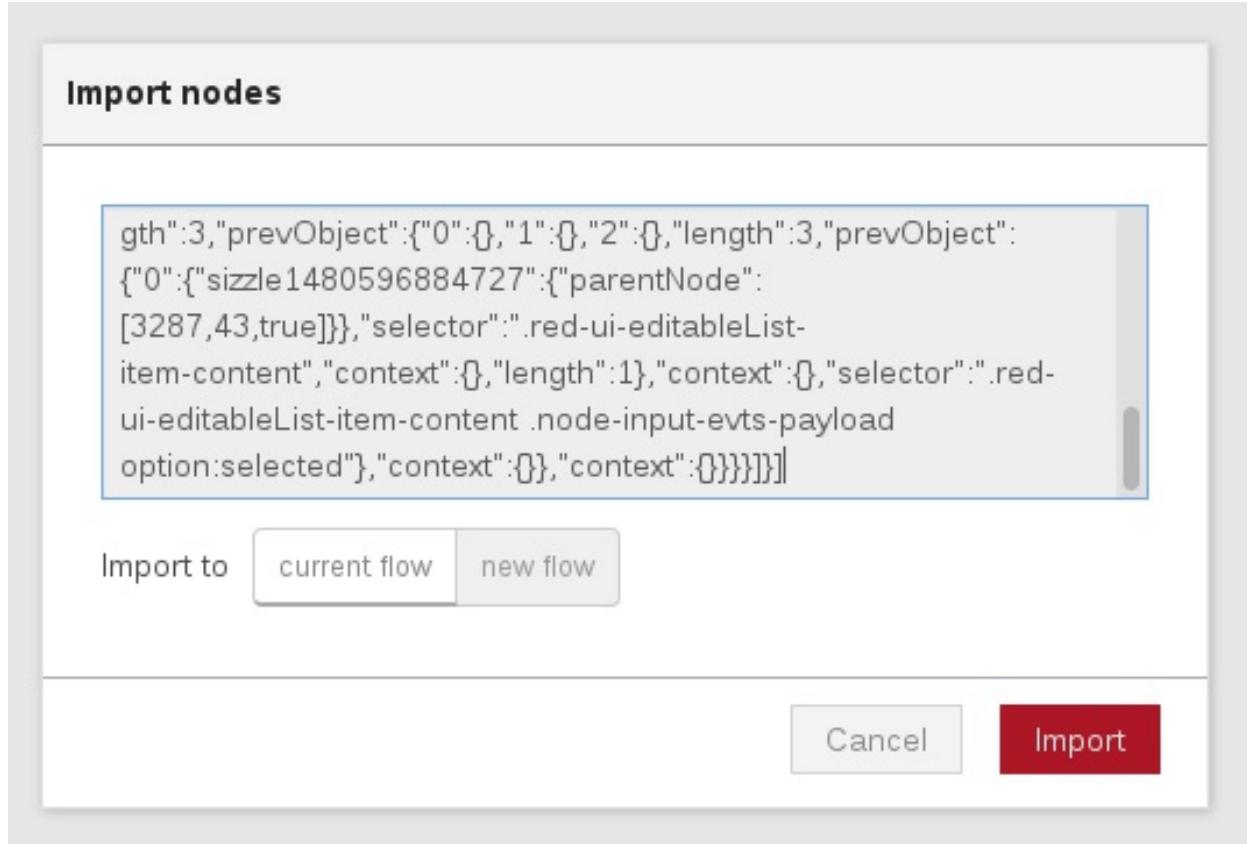
## Step 4 - How to import a prebuilt flow from GitHub

In this step, you will **learn** how to Import a prebuilt flow from GitHub

- Since configuring Node-RED nodes and wiring them together requires many steps to document in screenshots, there is an easier way to build a flow by importing a prebuilt flow into your IoT Starter Application.
- Not here in Step 4, but in several sections below, there will be a **Get the Code** link.
- When instructed in those later sections, open the Get the Code github URL, mark or Ctrl-A to select all of the text, and copy the text for the flow to your Clipboard.
- Click on the Node-RED Menu (6), then Import (7), then Clipboard (8).



- Paste the text of the flow into the **Import nodes** dialog and press the red **Import** button. This is an example of how to paste flows into your Node-RED editor. Don't type the text in the screenshot!



- The new flow will be imported into a new tab in the Node-RED Editor.
- Click the **Deploy** button on the top of menu bar to deploy the Node-RED flow.

## Congratulations - Node-RED is configured

Continue to the next step - [Sensor Data](#)

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Receive Device Environmental Sensor Data in Node-RED

## Lab Objectives

In this lab you will build a flow that receives Device environmental temperature and humidity sensor data. You will learn:

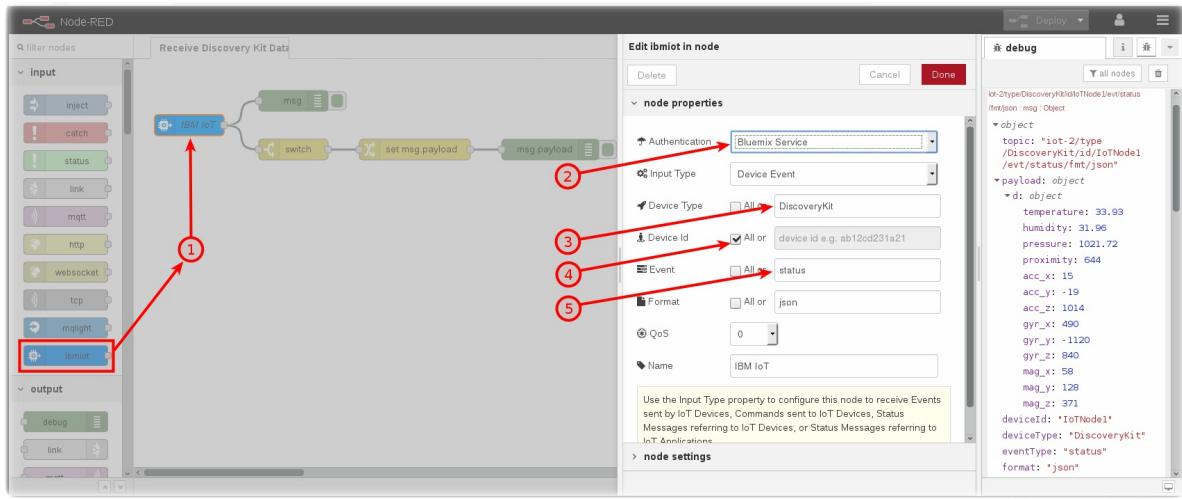
- How to create a new Node-RED flow and configure IoT Nodes
- How to output the Device environmental temperature and humidity data.
- How to work with JSON data and observe the sensor results in the Debug sidebar.

## Introduction

In just a few nodes, Node-RED can receive the data that was transmitted from the device over MQTT to Watson IoT Platform. This simple exercise will be the foundation for the next several sections that plot the data in a dashboard, trigger Real Time threshold alerts, store the data in Cloud Storage and allow for data analytics and anomaly detection.

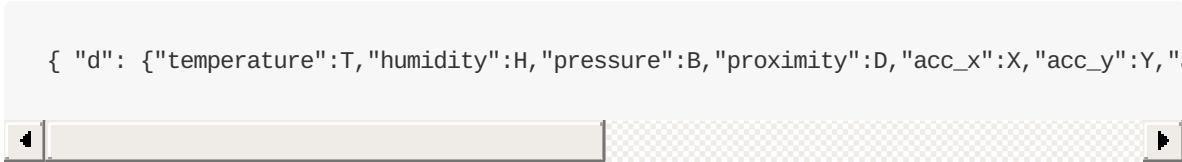
## Step 1 - Configure an IoT in Node

- From the Input category of the left Node-RED palette, select an **ibmiot node** and drag it onto your Node-RED flow (1).
- Double-click on the IBM IoT node. An **Edit ibmiot in node** sidebar will open.
- Configure the Authentication dropdown to **Bluemix Service** (2).
- Uncheck All and set the Device Type to **DiscoveryKit** (3).
- Check All Device Ids (4)
- Uncheck All and set the Event to **status** (5).
- Click on the red **Done** button.

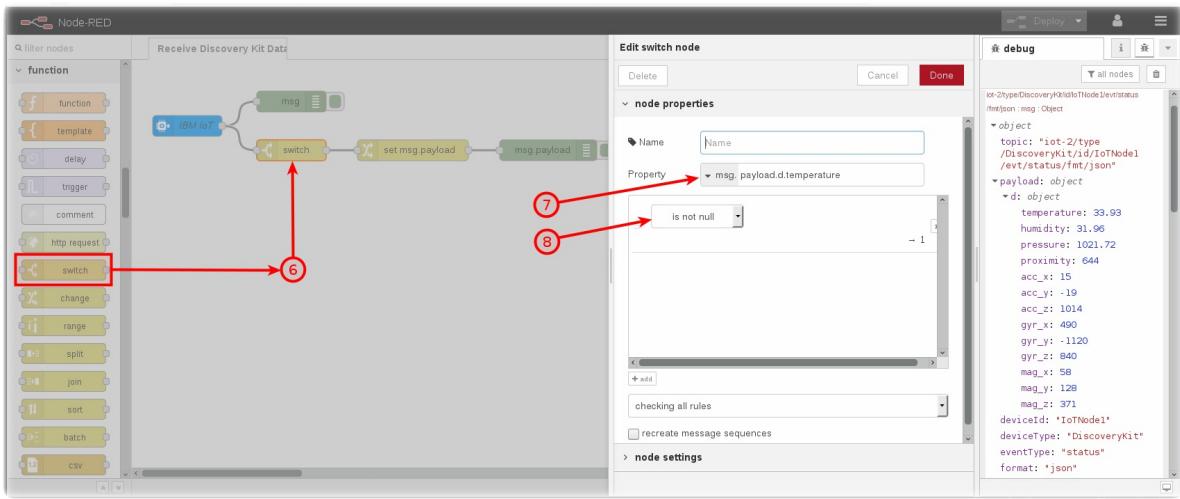


## Step 2 - Extract the Temperature from the JSON Object

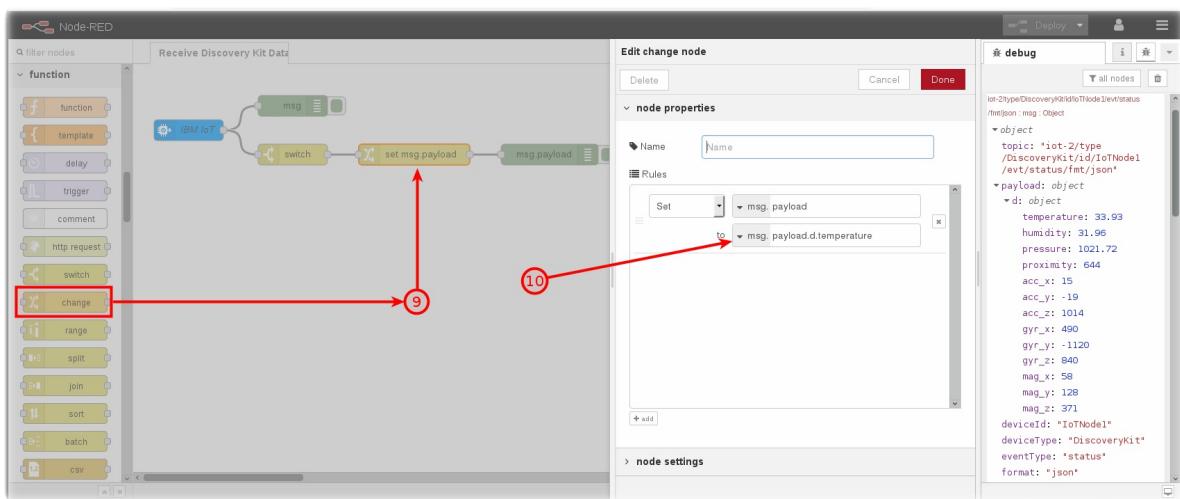
- Recall that the environmental sensor data was transmitted in a JSON object



- Node-RED passes data from node to node in a *msg.payload* JSON object.
- The **Switch** node can be used to test if the *msg.payload.d.temperature* is not null.
- From the Function category of the left Node-RED palette, select a **Switch** node and drag it onto your Node-RED flow (6).
- Double-click on the Switch node. An **Edit switch node** sidebar will open.
- Set the "Property" field to *payload.d.temperature* (7).
- From the dropdown, select **is not null** (8)
- Click on the red **Done** button.



- The **Change** node can be used to extract a particular value so that it can be directly output or manipulated (for instance in a Dashboard chart which we will take advantage of in the next section).
  - From the Function category of the left Node-RED palette, select a **Change** node and drag it onto your Node-RED flow (9).
  - Double-click on the Change node. An **Edit change node** sidebar will open.
  - Configure the "to" AZ dropdown to msg. and set it to *payload.d.temperature* (10).
  - Click on the red **Done** button.

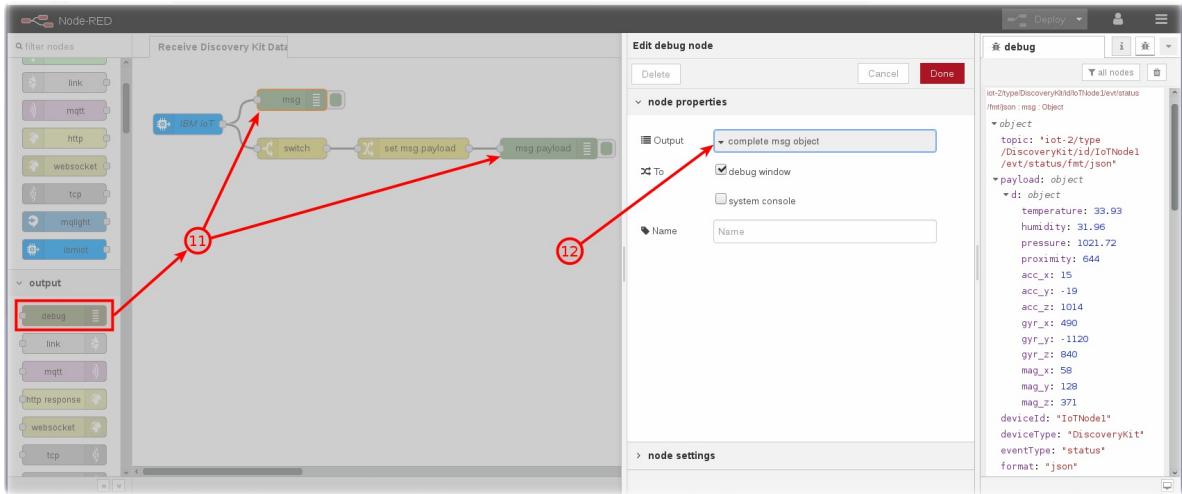


# Step 3 - Node-RED Debug Nodes

- Debug nodes can be used to print out JSON object values and help you validate your program.
  - From the Output category of the left Node-RED palette, drag two **debug nodes** onto your

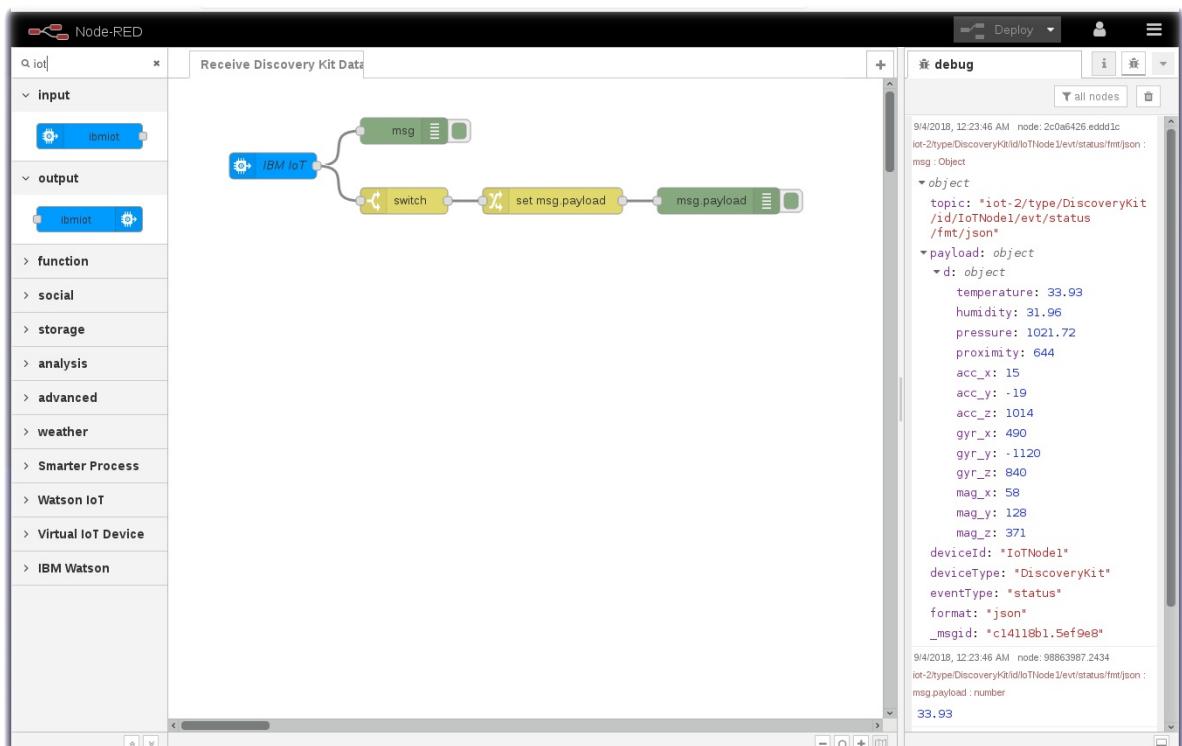
### Node-RED flow (11).

- Double-click on one of them. An **Edit debug node** sidebar will open.
- Configure the Output to print the *complete msg object* (12).
- Click on the red **Done** button.



## Step 4 - Wire the Node-RED nodes together

- Wire the Node-RED nodes together by click / dragging your mouse from nodelet to nodelet as show in the screenshot.
- Click on the red **Deploy** button in the upper right corner.
- Observe the DiscoveryKit IoT Node sensor data in the **debug** tab of the Node-RED right sidebar. You can expand the twisties to expose the JSON object information. Hover over a debug message in the right sidebar and the node that generated the message will be outlined in orange.



**Congratulations** - Your Node-RED flow is receiving sensor data from the Discovery Kit board.

Continue to the next step - [Node-RED Charts](#)

---

*Quick links :*

[Home - IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

---

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Node-RED Dashboard Charts - Plot Environmental Sensor Data

## Lab Objectives

In this lab you will import Node-RED flows which create Dashboard Charts. After learning about Node-RED Dashboard Charts, you will be able to display temperature and humidity graphs of the Device environmental sensors. You will learn:

- How to create a Node-RED Dashboard
- Experiment with Chart types
- Plot Real Time sensor data
- Trigger alerts when the real-time sensor data exceeds a threshold value

## Introduction

In this section you will learn about Node-RED Dashboard Charts and then create a chart to graph the sensor data arriving from the device.

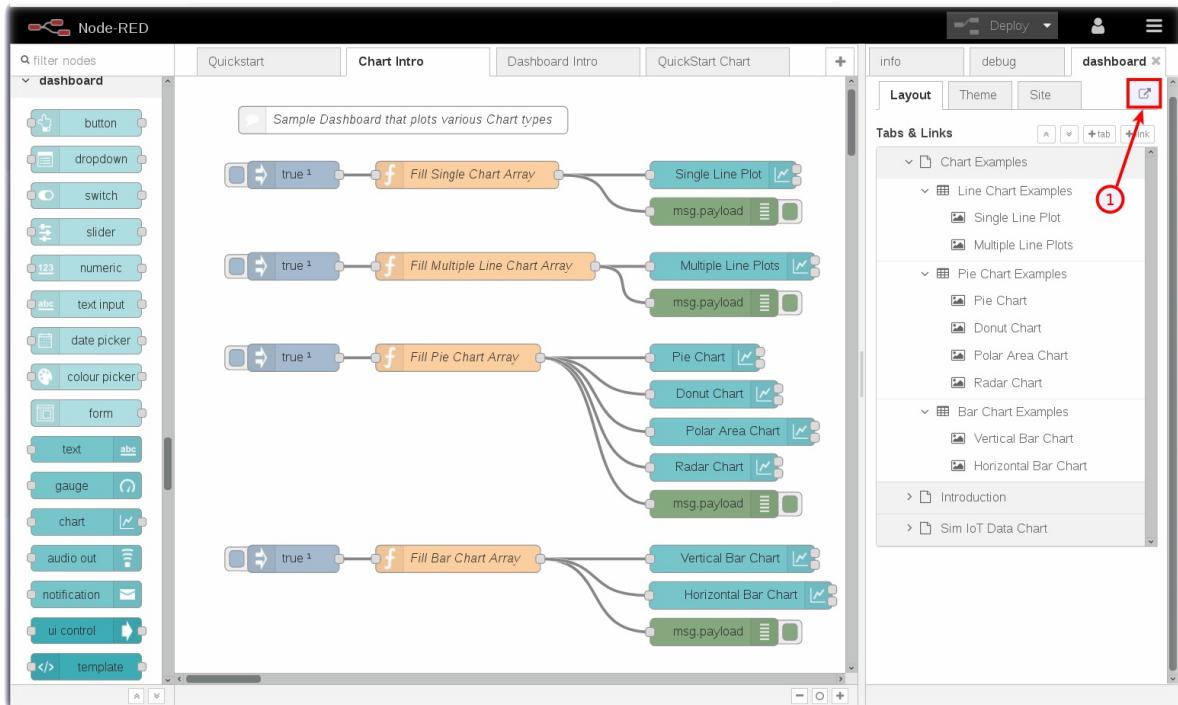
## Step 1 - Import the Node-RED Dashboard Chart Flows

Open the “Get the Code” github URL listed below, mark or Ctrl-A to select all of the text, and copy the text for the flow to your Clipboard. Recall from a previous section, click on the Node-RED Menu, then Import, then Clipboard. Paste the text of the flow into the Import nodes dialog and press the red Import button. Finally, click on the red Deploy button in the upper right corner.

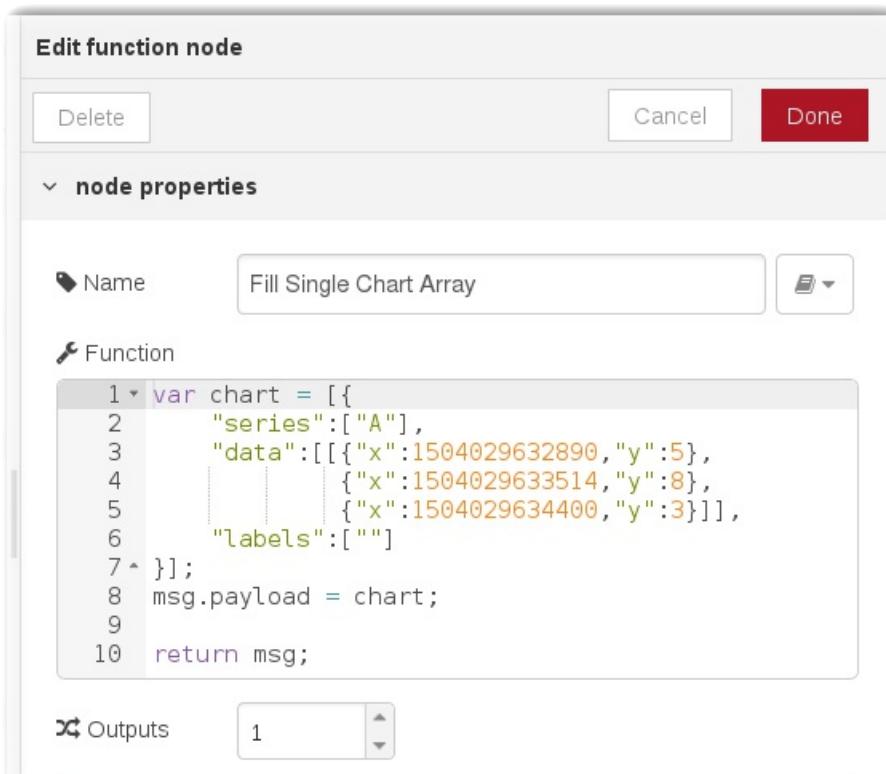
Get the Code: [Node-RED Dashboard Charts](#)

## Step 2 - Learn about various Node-RED Dashboard Chart types

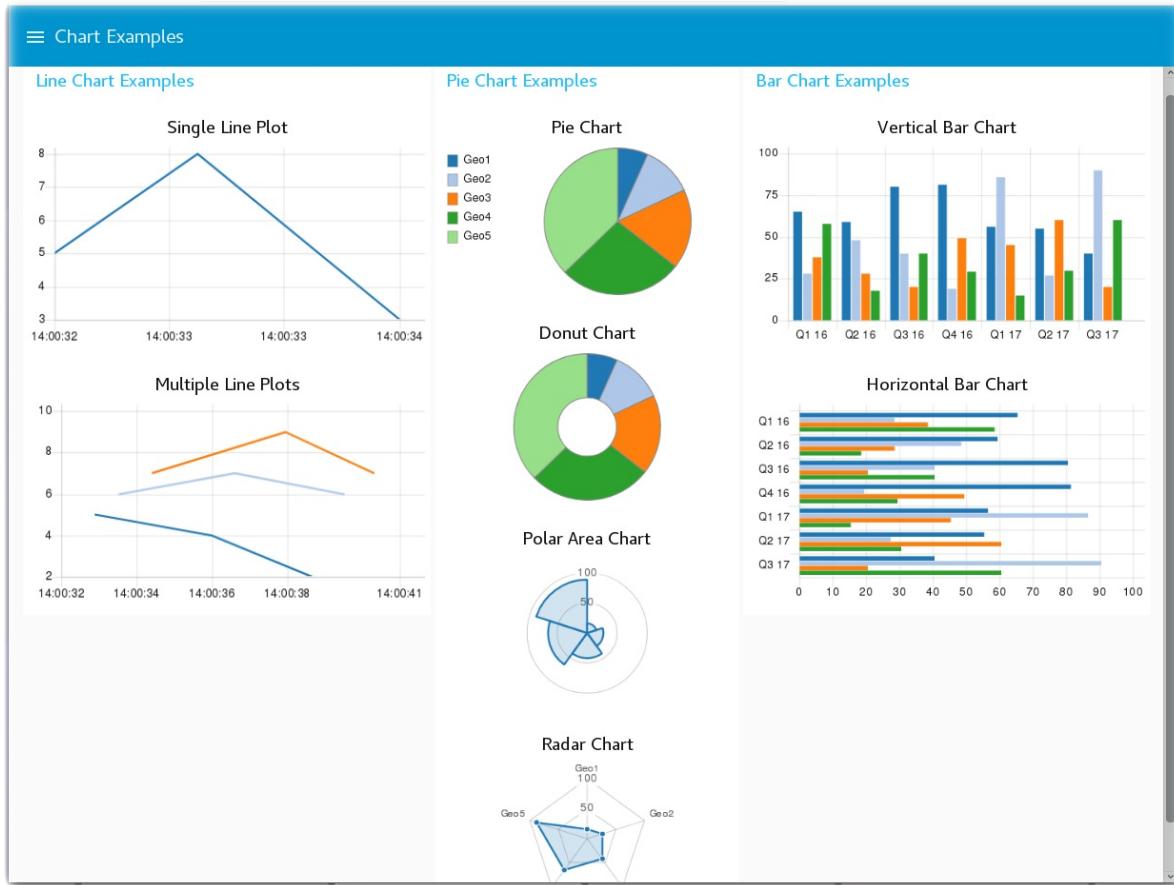
- You might have noticed that the flow import in the prior step actually created three tabs. These are called Node-RED flows.
- The **Chart Intro** flow introduces you to the various Node-RED Dashboard Chart types that are available. You can create Line charts, various Pie chart styles – Pie, Donut, Polar Area, Radar - and vertical and horizontal Bar charts.



- For illustration, the **Fill Single Chart Array** function node above fills an array of static sample data and sends the data to the **Chart** node to visualize.

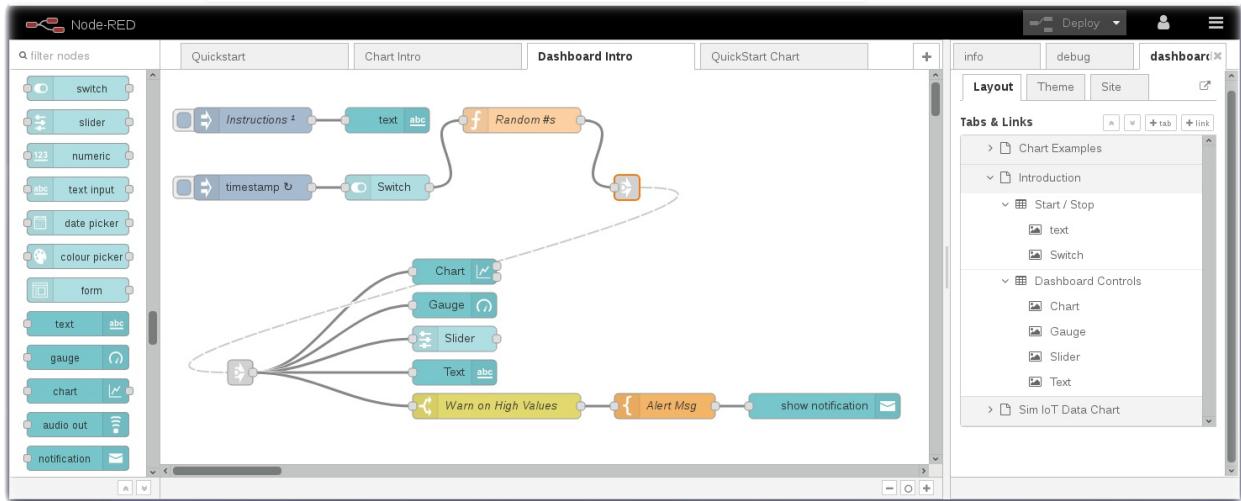


- The Node-RED flow is not nearly as interesting as the charts that it renders. To launch the Node-RED Dashboard, in the Node-RED sidebar, turn to the **dashboard** tab and click on the **Launch** button (1) [See two pictures above].

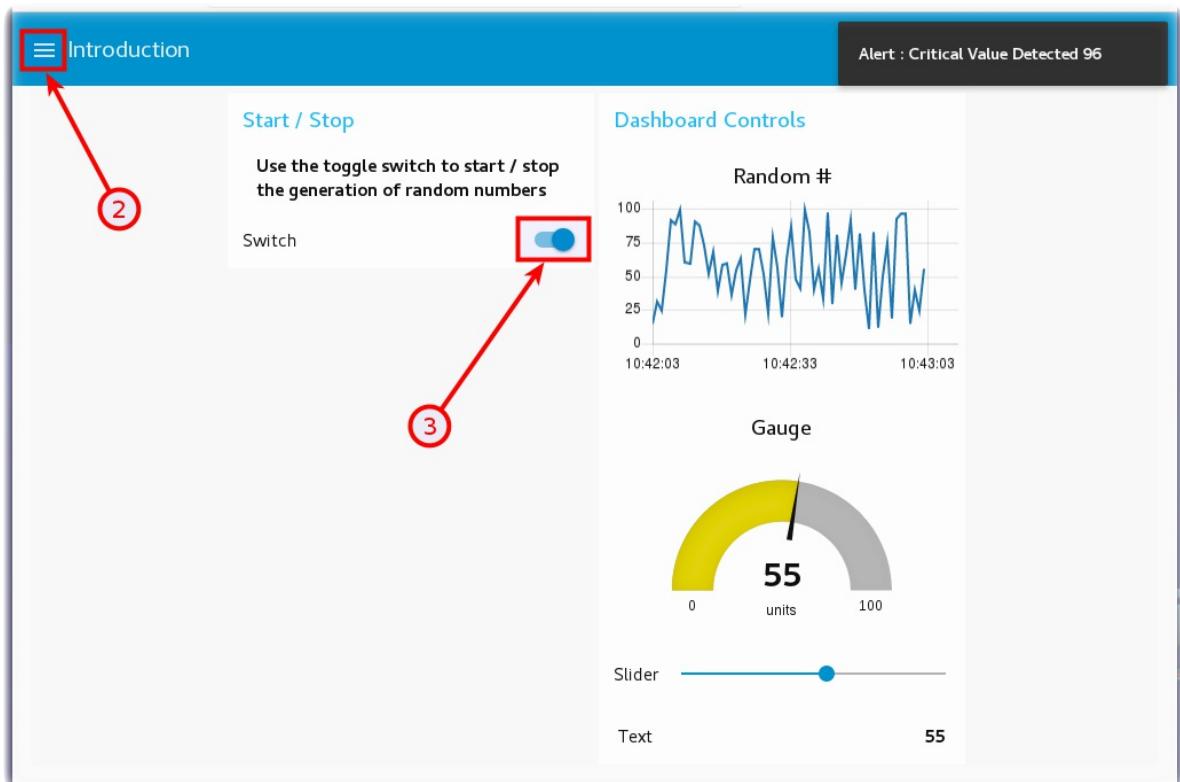


## Step 3 – Generating and Displaying data in Node-RED Dashboards

The next Node-RED flow - **Dashboard Intro** - uses a variety of UI widgets to display data in the Node-RED Dashboard. There is a Switch node that turns On/Off a random number generator function node. The simple random numbers are sent to a line Chart node, a Gauge node, a Slider node, a Text node and, if the number exceeds a threshold, will display an alert notification message.



- Turn to the Node-RED Dashboard browser tab that you launched in Step 2, click on the menu tab (2) in the upper left corner, and select the Introduction tab.
- On the Introduction dashboard, turn on the **Switch** (3) to start the data visualization.
- Experiment with / observe the Dashboard controls.



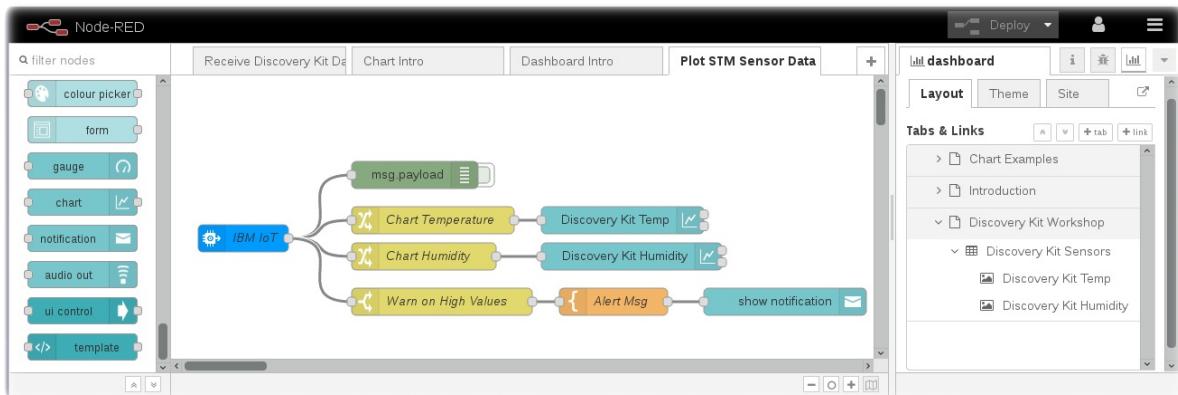
## Step 4 - Plot Device Environmental Sensor Data

Now that you have learned about Node-RED Dashboard and Chart types, you are ready to plot the real-time device environmental sensor data.

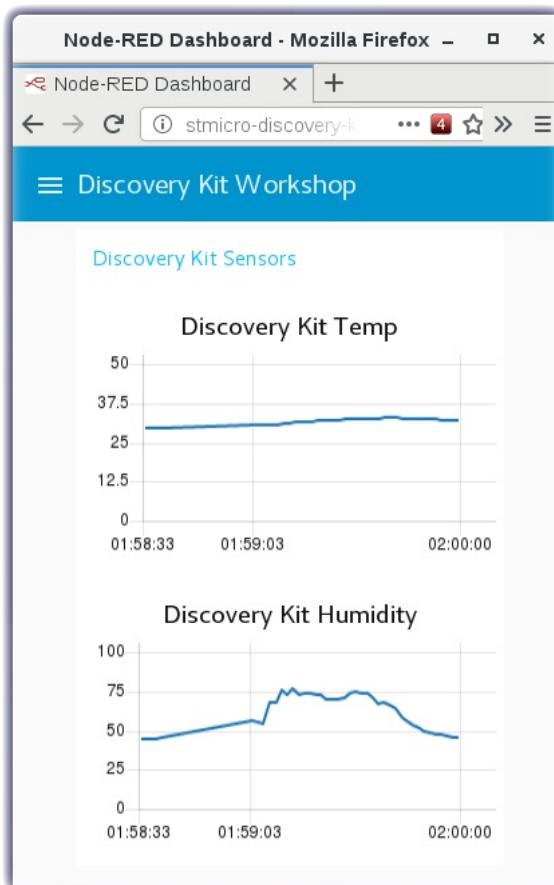
- Turn to the next flow - **Plot STM Sensor Data**
- The **IBM IoT** node is already configured to receive *status* Device Events from the

DiscoveryKit Device Type.

- The **Change** nodes extract the `msg.payload.d.temperature` and `msg.payload.d.humidity` values from the JSON object sent over MQTT from the device environmental sensor to Watson IoT Platform.
- The environmental sensor values are sent to two charts to plot Temperature and Humidity.



- Turn to the Node-RED Dashboard browser tab that you launched in Step 2, click on the tab in the upper left corner, and select the **Discovery Kit Workshop** tab.

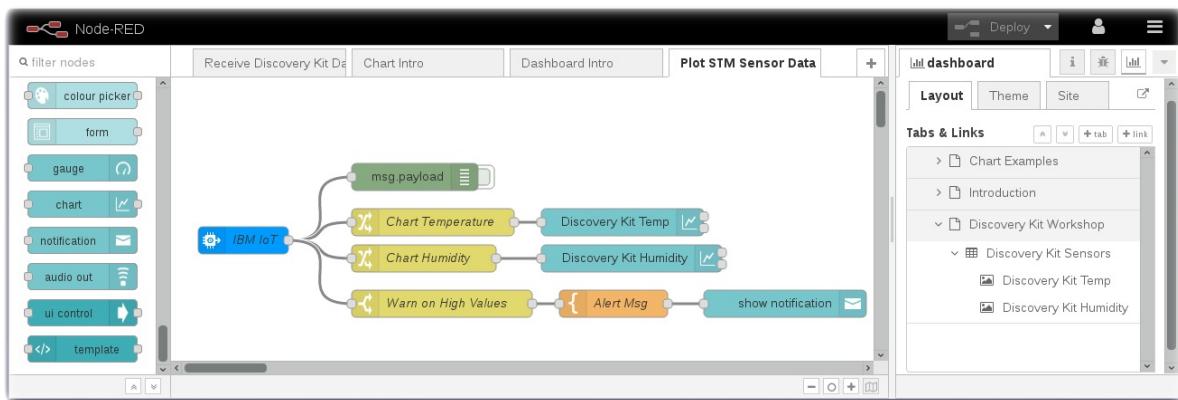


## Step 5 - Trigger Alerts when Real-Time Sensor Data Exceeds a Threshold Value

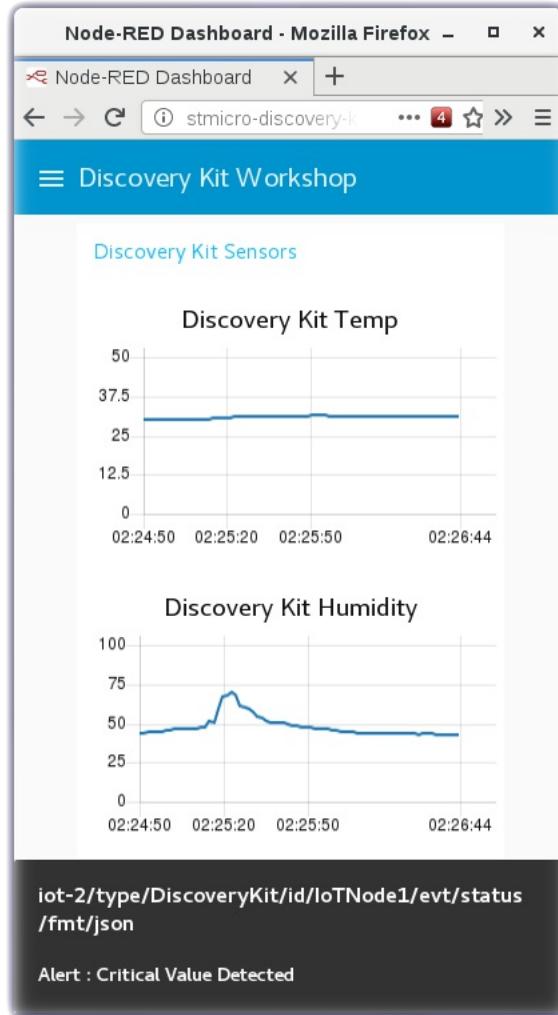
Often IoT devices and sensors are deployed so that alerts can be triggered when the real time

sensor data exceeds a threshold value. In this last Step, the flow checks the temperature values and, if the temperature exceeds the threshold, triggers a Node-RED Dashboard notification.

- In the prior step, the flow included three nodes that have not yet been discussed.
- A **Switch** node is configured to *Warn on High Values* by testing if `msg.payload.d.temperature` is greater than 30C.
- A **Template** node is configured to construct a sentence `Alert : Critical Value Detected {{payload.d.temperature}}`
- The Alert message is sent to a **Node-RED Dashboard Notification** node to display in the browser.
- This flow could be extended to call a **Twilio** node to send a SMS message. It could raise an alarm in another system by triggering a REST API call to the manufacturing production operations systems.



- Return to the Node-RED Dashboard **Discovery Kit Workshop** tab and increase the temperature of your Discovery Kit sensor above 30C.



*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Store Data in Cloud Storage for Historical Data Analytics

---

## Lab Objectives

---

In this lab you will store the device environmental sensor data in a Cloudant database in IBM Cloud. You will learn:

- How to create a Node-RED flow that uses the Cloudant node
- How to format a time series database record
- How to view Cloudant databases

## Introduction

While real-time charts of sensor data and threshold alerts are useful, the power of IoT becomes significant when data analytics techniques, Machine Learning and AI are applied to the IoT historical datasets. The first and necessary step toward data analytics is storing the incoming data in a Cloud data storage database for later statistical modelling.

## Step 1 - Import the Node-RED Cloudant Storage Flow

Open the “Get the Code” github URL listed below, mark or Ctrl-A to select all of the text, and copy the text for the flow to your Clipboard. Recall from a previous section, click on the Node-RED Menu, then Import, then Clipboard. Paste the text of the flow into the Import nodes dialog and press the red Import button.

[Get the Code: Node-RED Cloud Storage Flow](#)

## Step 2 - Store IoT Sensor Data with Node-RED

In this Step you will use Node-RED to store IoT Sensor data from the ST Microelectronics Discovery Kit IoT Node environmental sensors in a Cloudant database.

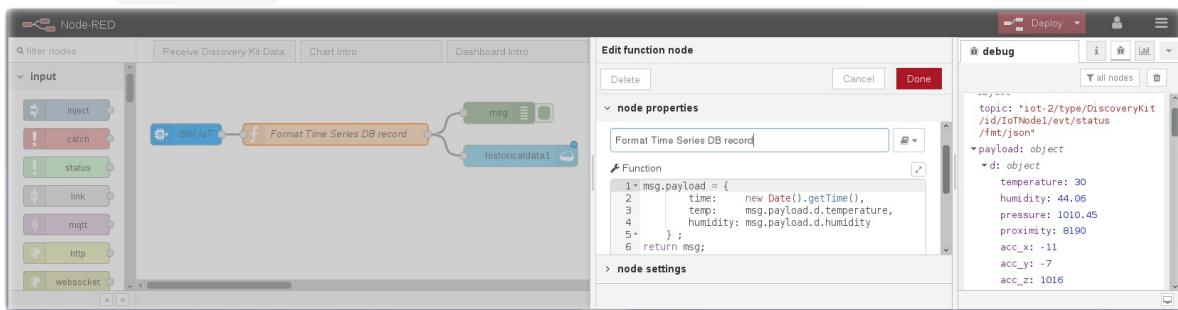
- When the flow is imported there will be a misconfigured Cloudant node – indicated by a red triangle.



- To associate the **Cloudant** database node with your IBM Cloud instance, click on the historical data Cloudant node and press the red Done button. The red error triangle will turn blue.



- The *Format Time Series DB Record* function node recasts the Discovery Kit JSON object. As required by any time series dataset, the Node-RED function node adds a timestamp to the record before writing it to the Cloudant storage. Note in the screenshot, the debug sidebar shows a `msg.payload` that includes the Epoch timestamp (milliseconds since Jan 1 1970)



- Click the **Deploy** button on the top of menu bar to deploy the Node-RED flow.
- The device environmental sensor data is now being recorded in a Cloudant database.

## Step 3 - Observe Sensor Data being added to the Cloudant database

- Return to the [IBM Cloud dashboard](#) and your IoT Starter application. Click on the cloudantNoSQLDB (1) service connection.

The screenshot shows the IBM Cloud application overview page for 'stmicro-discovery-kit-iot-node'. The left sidebar includes links for Getting started, Overview, Runtime, Connections, Logs, Monitoring, and API Management. The main area displays the application name, status (Running), and a 'Visit App URL' button. It also shows the organization (walicki@us.ibm.com), location (US South), and space (dev). A 'Runtime' section provides details: Buildpack (.js, SDK for Node.js™), Instances (1, healthy), MB Memory per Instance (256), and Total MB Allocation (256, 12.625 GB available). Below this, a 'Connections' section lists two entries: 'stmicro-discovery-kit-iot-node-cloudantNoSQL...' and 'stmicro-discovery-kit-iot-node-iotf-service'. A red arrow points from a circled '1' in the sidebar to the 'Connections' section.

- Click on the *Alias of Cloudant-NNN* link

The screenshot shows the connection details for 'stmicro-discovery-kit-iot-node-cloudantNoSQLDB'. The left sidebar has 'Connections' selected. The main area shows the connection name, location (US South), organization (walicki@us.ibm.com), and space (dev). A red box highlights the 'Alias of Cloudant-NNN' link. Below this, a 'Connected Applications' table lists one entry: 'stmicro-discovery-kit-iot-node' connected to 'stmicro-discovery-kit-iot-node.mybluemix.net' with a status of 'Running'.

- Read about the Cloudant Storage service and press the **Launch** button.

The screenshot shows the Cloudant service page. The left sidebar has 'Manage' selected. The main area displays the Cloudant service details: Data & Analytics / Cloudant-ld, Resource Group: default, Location: US South. A 'Cloudant' section contains a 'LAUNCH CLOUDANT DASHBOARD' button. Below this, there's information about Cloudant being a fully managed JSON document database compatible with Apache CouchDB, and a note about its throughput capacity starting at 100 lookups/sec, 50 writes/sec, and 5 queries/sec. The 'Plans' section notes that Cloudant has a free Lite plan and a paid Standard plan.

- The IoT Sensor device data is stored in the Cloudant service.

The screenshot shows the 'Databases' section of the Watson Studio interface. It lists two databases: 'historicaldata1' and 'nodered'. The 'historicaldata1' database has a size of 68.8 KB and 206 documents. The 'nodered' database has a size of 116.8 KB and 3 documents. Each database row includes action buttons for managing the database.

Name	Size	# of Docs	Actions
historicaldata1	68.8 KB	206	
nodered	116.8 KB	3	

Showing 1–2 of 2 databases. | « | 1 | » |

- Click on historicaldata1 and then observe the **Table** view of temperature, humidity and timestamp data.

The screenshot shows the 'historicaldata1' database table view. The table displays sensor data with columns: \_id, humidity, temp, and time. The data consists of 8 rows of sensor readings.

_id	humidity	temp	time
20e2e13a56fa66...	40.5	22.6	1523756677898
20e2e13a56fa66...	31.6	27.6	1523756791107
20e2e13a56fa66...	29.6	28.8	1523756811629
20e2e13a56fa66...	26.4	31.5	1523756852779
20e2e13a56fa66...	31.2	25.5	1523757458986
20e2e13a56fa66...	34.6	24	1523757613285
2a230ede8edf3fa...	34.9	25.9	1523756760196

Showing 4 of 5 columns.  Show all columns. | Documents per page: 20 | < | > |

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Node-RED Charts of Historical Sensor Data

---

## Lab Objectives

---

In this lab you will read the historical sensor data from a Cloud storage database and create a graph of prior readings. You will learn:

- How to read datasets from a Cloudant database
- How to create a chart of historical data

## Introduction

The previous section stored the Device environment sensor data into a Cloudant DB. This section will read the historical sensor data from a Cloud storage database and create a graph of prior readings.

## Step 1 - Import the Node-RED Historian Chart Flow

- Open the “Get the Code” github URL listed below, mark or Ctrl-A to select all of the text, and copy the text for the flow to your Clipboard. Recall from a previous section, click on the Node-RED Menu, then Import, then Clipboard. Paste the text of the flow into the Import nodes dialog and press the red Import button.

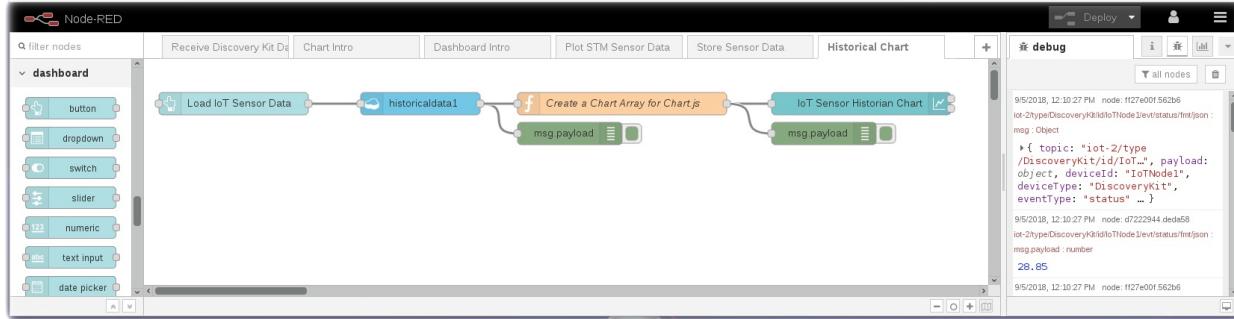
[Get the Code: Node-RED Historian Chart Flow](#)

- Click on the **Cloudant** node on the Historical Chart flow to confirm that it is configured to your IoT Platform Cloudant service instance.
- Click the **Deploy** button on the top of menu bar to deploy the Node-RED flow.

## Step 2 - Graph Historical IoT Sensor data stored in a

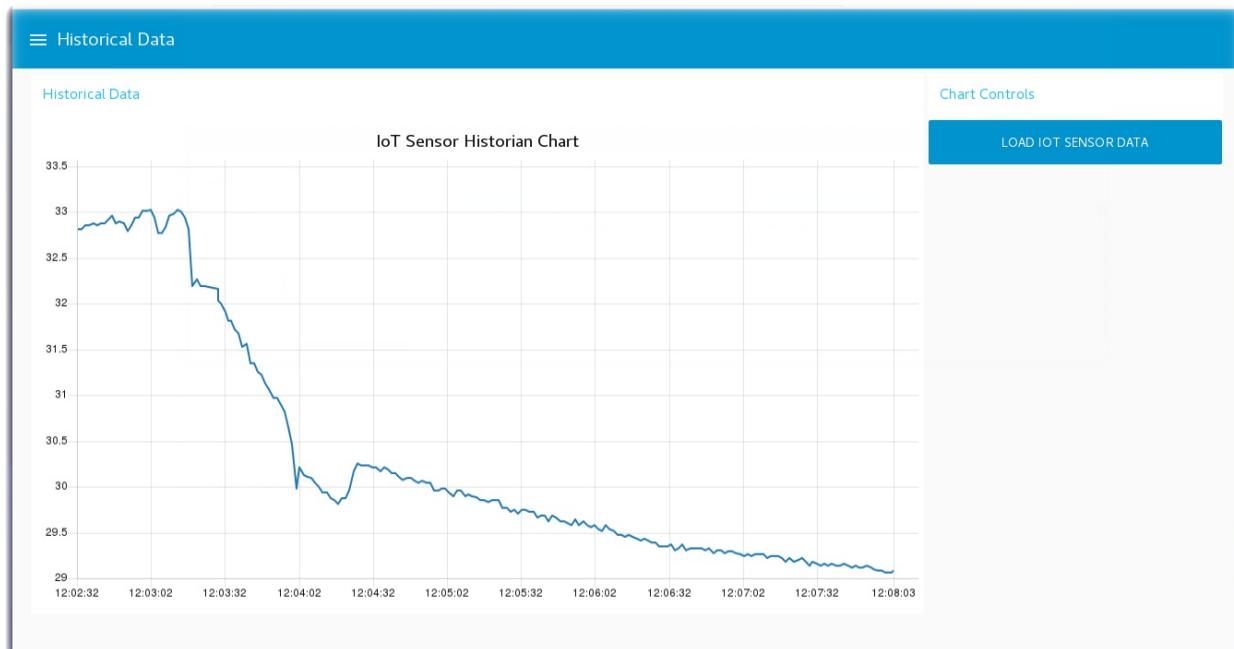
# database using Node-RED

- The Historical Chart flow reads the IoT Sensor Device data from the Cloudant database and formats it into a Chart array before sending the data to a Node-RED Chart node.



## Step 3 - Historian Charts of Device Environmental Sensor data

- Turn to the Node-RED Dashboard browser tab, click on the menu tab in the upper left corner, and select the Historical Data tab.
- On the Historical Data dashboard, click on the **LOAD DISCOVERY KIT DATA** button to start the data visualization.
- The button will trigger the read of the historian DB records created in the previous section.
- In the *Create a Chart Array for Chart.js* function node the time series temperature data from the Device Environmental sensor is formatted into a Chart Array and sorted chronologically.
- The Chart Array is passed to the Node-RED Chart node to render the graph.



*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Watson Studio Set up and Configuration in IBM Cloud

## Lab Objectives

In this lab you will set up Watson Studio with a new Project. You will learn:

- Watson Studio
- How to set up a new Watson Studio Project
- How to create a Jupyter Notebook

## Introduction

Watson Studio accelerates the machine and deep learning workflows required to infuse AI into your business to drive innovation. It provides a suite of tools for data scientists, application developers and subject matter experts, allowing them to collaboratively connect to data, wrangle that data and use it to build, train and deploy models at scale. Successful AI projects require a combination of algorithms + data + team, and a very powerful compute infrastructure.

- Learn more from the Experts - [Introducing IBM Watson Studio](#)

## Step 1 - Watson Studio Setup

### Create Cloud Object Storage

- Create a Cloud Object Storage instance by visiting the [IBM Cloud Catalog](#)
- Search on **Object** in the IBM Cloud Catalog
- Click on the **Object Storage** service tile

The screenshot shows the IBM Cloud Catalog interface. At the top, there's a navigation bar with links for Catalog, Docs, Support, and Manage. A search bar and user information (1421965 - John Walick...) are also at the top. Below the navigation is a search bar with the placeholder "object" and a "Filter" button. A sidebar on the left shows "All Categories (6)" with links to Compute, Containers, Networking, Storage (1), AI (1), Analytics (2), Databases (2), Developer Tools, Integration, Internet of Things, Security and Identity, Starter Kits, Web and Mobile, and Web and Application. The main content area is titled "Storage" and contains two service cards: "Object Storage" (Lite • IBM) which provides flexible, cost-effective, and scalable cloud storage for unstructured data; and "Visual Recognition" (Lite • IBM) which finds meaning in visual content by analyzing images for scenes, objects, faces, and other content. A "FEEDBACK" button is located on the right side of the main content area.

- Click on the **Create** button

The screenshot shows the details page for the "Cloud Object Storage" service. At the top, there's a navigation bar with links for Catalog, Docs, Support, and Manage. A search bar and user information (1421965 - John Walick...) are also at the top. Below the navigation is a "View all" link and the service name "Cloud Object Storage" with a "Lite • IBM" badge. To the right, there's a "Compare Versions" button and a "FEEDBACK" button. The main content area includes a description of the service, which is highly scalable and designed for durability, resiliency, and security. It allows users to store, manage, and access their data via a self-service portal and RESTful APIs. It connects applications directly to Cloud Object Storage using other IBM Cloud Services. There are also "View Docs" and "Terms" links. On the left, there are metadata fields: AUTHOR (IBM), PUBLISHED (06/25/2018), and TYPE (Service). In the center, there are sections for "Service name:" (containing "Cloud Object Storage-jw"), "Select a resource group:" (containing "default"), and "Features". The "Features" section lists "Storage for the IBM Cloud" (described as providing unstructured data storage for cloud applications) and "IAM Policies - Bucket level access management" (described as allowing granular access control at the bucket level using role-based policies). At the bottom, there are links for "Need Help?", "Contact IBM Cloud Support", "Estimate Monthly Cost", and "Cost Calculator", along with a prominent "Create" button.

## Create a Watson Studio service instance

- Create a **Watson Studio** service instance from the [IBM Cloud Catalog](#)
- Search on **Studio** in the IBM Cloud Catalog

IBM Cloud Catalog Docs Support Manage

Catalog 1421965 - John Walick...

**Catalog**

Studio Filter

All Categories (4) > AI

Compute  
Containers  
Networking  
Storage  
AI (3)  
Analytics (1)  
Databases  
Developer Tools  
Integration  
Internet of Things  
Security and Identity  
Starter Kits

<https://console.bluemix.net/catalog/services/watson-studio>

**Knowledge Studio** Lite • IBM  
Build custom models to teach Watson the language of your domain.

**Natural Language Understanding** Lite • IBM  
Analyze text to extract meta-data from content such as concepts, entities, emotion, relations, sentiment and more.

**Watson Studio** Lite • IBM  
Embed AI and machine learning into your business. Create custom models using your own data.

FEEDBACK

- Click on the **Watson Studio** service tile

IBM Cloud Catalog Docs Support Manage

View all **Watson Studio** Lite • IBM

Watson Studio democratizes machine learning and deep learning to accelerate infusion of AI in your business to drive innovation. Watson Studio provides a suite of tools and a collaborative environment for data scientists, developers and domain experts.

Service name: Watson Studio-jk

Choose a region/location to deploy in: US South Select a resource group: default

View Docs Terms

AUTHOR IBM  
PUBLISHED 08/01/2018  
TYPE Service

Features

• Use what you know, learn what you don't Start from a tutorial, start from a sample, or start from scratch. Tap into the power of the best of open source (RStudio, Jupyter Notebooks) and Watson services for flexible model creation. Use Python, R, or Scala. Stop downloading and configuring analysis environments and start getting insights.

• Power on demand Enterprise-scale features on demand. From data exploration and preparation, to enterprise-scale performance. Manage your data, your analytical assets, and your projects in a secured cloud environment.

Need Help? Contact IBM Cloud Support Estimate Monthly Cost Cost Calculator Create

- Click on the **Create** button
- After the Watson Studio service is created, click on **Get Started** or visit Watson Studio at <http://dataplatform.ibm.com>

The screenshot shows the IBM Cloud Watson Studio interface. At the top, there's a navigation bar with 'IBM Cloud' and links for 'Catalog', 'Docs', 'Support', and 'Manage'. On the right, there's a search bar and a user profile icon. The main area has a left sidebar with 'Manage' and 'Plan' sections. The main content area displays 'Watson / Watson Studio-ge' with a resource group of 'default' and location 'US South'. It features a purple icon of a desk lamp and a 'Watson Studio' title. Below it, a message says 'Welcome to Watson Studio. Let's get started!' with a 'Get Started' button.

- Login with your IBM Cloud account
- Walk through the introductory tutorial to learn about Watson Studio

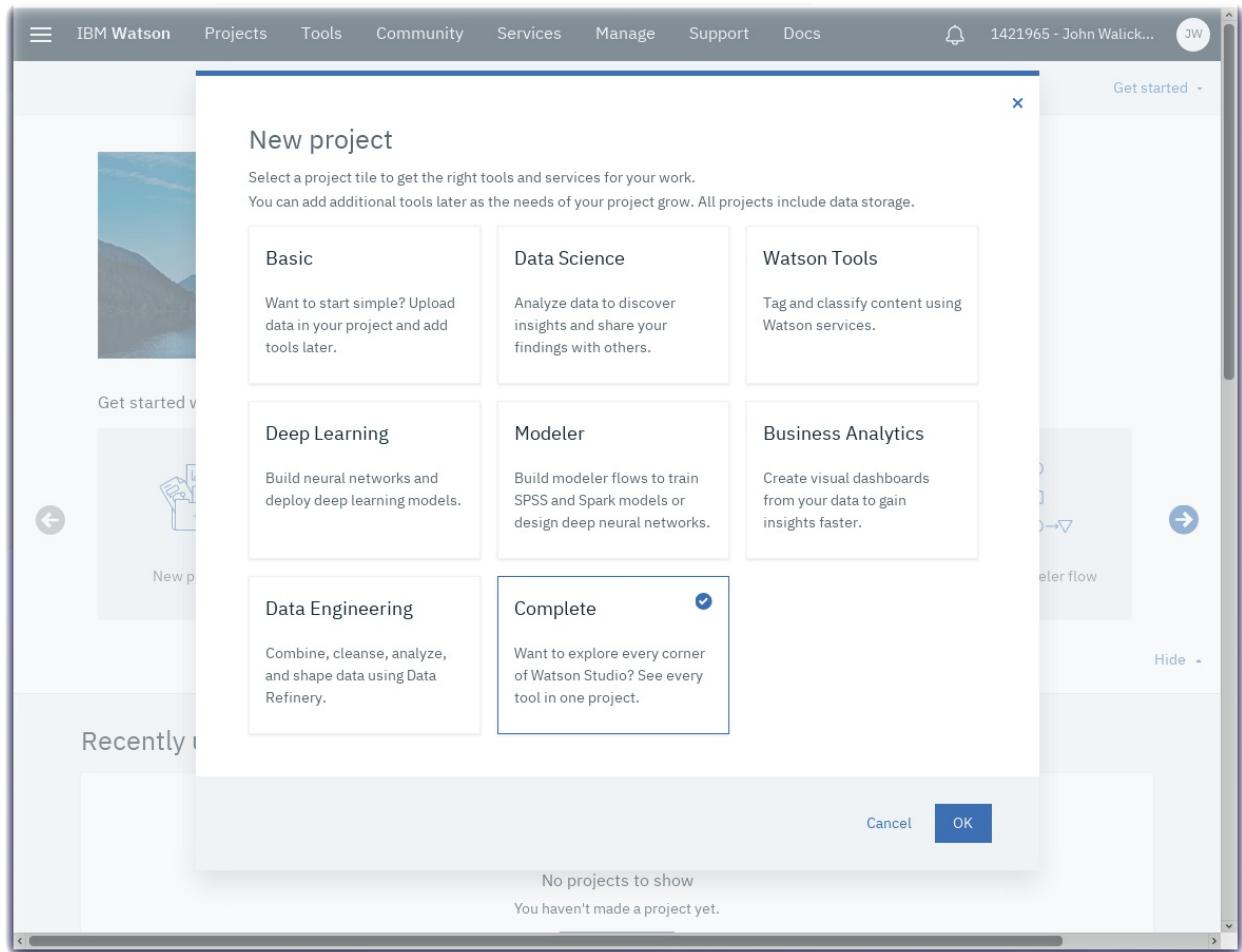
The screenshot shows the IBM Watson Watson Studio interface. At the top, there's a navigation bar with 'IBM Watson' and links for 'Projects', 'Tools', 'Community', 'Services', 'Manage', 'Support', and 'Docs'. On the right, there's a notification bell and a user profile icon. The main area has a large image of a lake with mountains and the text 'IBM Watson'. To the right, it says 'Welcome John!', 'Watson Studio is part of IBM Watson.', and 'Try out other IBM Watson apps.' Below this, there's a section titled 'Get started with key tasks' with six cards: 'New project' (document icon), 'Refine data' (database icon), 'New notebook' (notebook icon), 'Deep learning' (neuron icon), 'New Modeler flow' (flowchart icon), and 'New model' (diamond icon).

## Step 2 - Create a New Project

Projects are your workspace to organize your resources, such as assets like data, collaborators, and analytic tools like notebooks and models

### Create a New Project

- Click on **New project**
- Select the **Complete** tile and press the **OK** button



- Give your Project a name : **IoT Sensor Analytics**
- The Cloud Object Storage instance created in Step 1 should be prefilled
- Press the **Create** button

The screenshot shows the 'Define project details' step of the 'New project' dialog box. It is divided into two main sections: 'Define project details' on the left and 'Storage' on the right.

**Define project details:**

- Name:** IoT Sensor Analytics
- Description:** A large text input field with placeholder text 'Project description'.

**Storage:** A list showing 'Cloud Object Storage-40'.

**Choose project options:**

- Restrict who can be a collaborator (i)

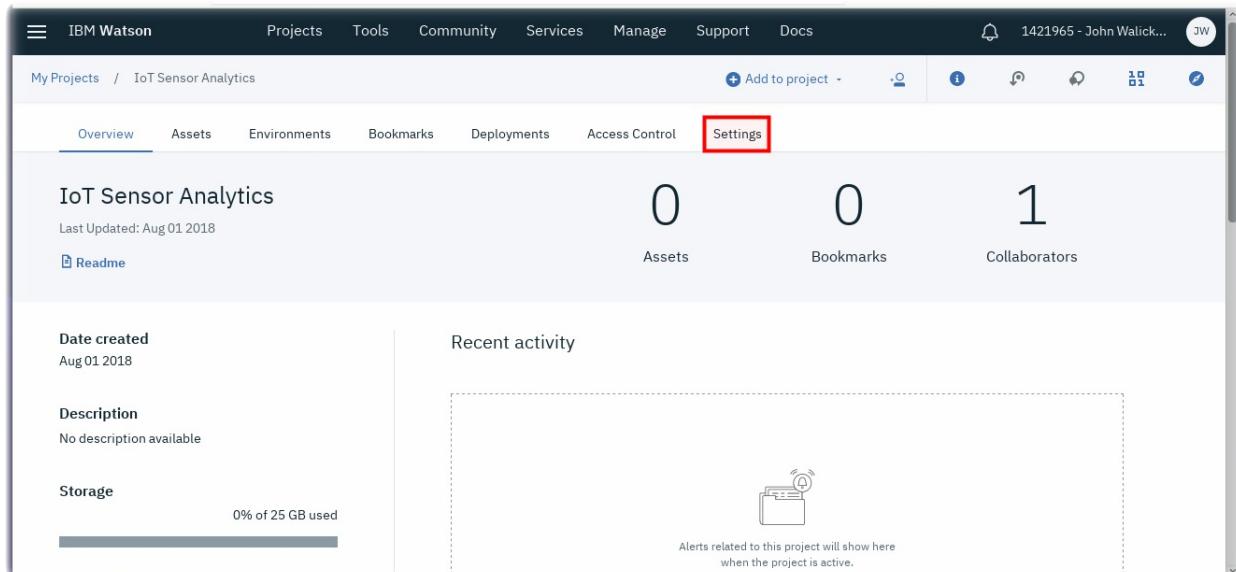
Project will include integration with [Cloud Object Storage](#) for storing project assets.

At the bottom right are 'Cancel' and 'Create' buttons.

# Add an Apache Spark Service

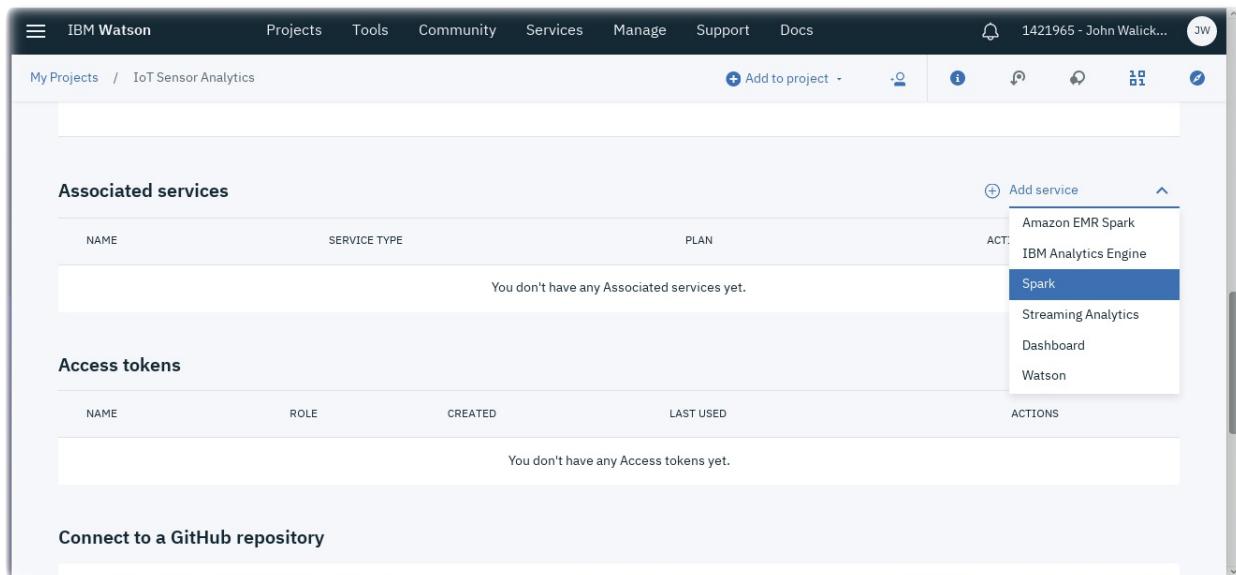
First add an Apache Spark service to the project.

- Press the **Settings** tab



The screenshot shows the IBM Watson interface for a project named "IoT Sensor Analytics". The top navigation bar includes links for IBM Watson, Projects, Tools, Community, Services, Manage, Support, and Docs. The user is logged in as "1421965 - John Walick...". Below the navigation is a breadcrumb trail: "My Projects / IoT Sensor Analytics". A toolbar with icons for adding to project, sharing, and more is visible. The main content area has tabs for Overview, Assets, Environments, Bookmarks, Deployments, Access Control, and Settings, with the Settings tab highlighted by a red box. The Overview section displays statistics: 0 Assets, 0 Bookmarks, and 1 Collaborator. It also shows project details like Date created (Aug 01 2018) and Description (No description available). The Storage section shows 0% of 25 GB used. To the right, there's a "Recent activity" panel with a placeholder message: "Alerts related to this project will show here when the project is active." A small icon of a folder with a signal is shown.

- Scroll down to the **Associated services** section and click on the **Add service** dropdown
- Select **Spark** from the dropdown



The screenshot shows the "Associated services" section within the project settings. The "Associated services" heading is at the top, followed by a table with columns: NAME, SERVICE TYPE, and PLAN. A message below the table says, "You don't have any Associated services yet." To the right of the table is an "Add service" dropdown menu. The menu is open, showing a list of service options: Amazon EMR Spark, IBM Analytics Engine, Spark (which is highlighted in blue), Streaming Analytics, Dashboard, and Watson. The "Spark" option is currently selected. Below the dropdown is a "Connect to a GitHub repository" section.

- In the **Apache Spark** service panel, select the **Lite** plan and press the **Create** button. Press the **Confirm** button.

The screenshot shows the IBM Watson interface for creating a new Apache Spark service. At the top, there's a navigation bar with links for Projects, Tools, Community, Services, Manage, Support, and Docs. On the right, there's a user profile and a notification bell. Below the navigation, the title 'Apache Spark' is displayed, with tabs for 'Existing' and 'New'. The 'New' tab is selected.

**Apache Spark**

Apache Spark is an open source cluster computing framework optimized for extremely fast and large scale data processing, which you can access via the newly integrated notebook interface IBM Analytics for Apache Spark. You can connect to your existing data sources or take advantage of the on-demand big data optimization of Object Storage. Spark plans are based on the maximum number of executors available to process your analytic jobs. Executors exist only as long as they're needed for processing, so you're charged only for processing done.

**Features**

- Incredibly Fast**: Apache Spark delivers 100x the performance of Apache Hadoop for certain workloads because of its advanced in-memory computing engine.
- Easy to Use and Powerful**: Apache Spark's Streaming and SQL programming models backed by MLlib and GraphX make it incredibly easy for developers and data scientists to build apps that exploit machine learning and graph analytics. Because the service is 100% compatible with Apache Spark, developers can build their apps and run them against the IBM managed service to benefit from operational, maintenance, and hardware excellence.
- Convenient Data Storage**: Object Storage enables a convenient way to upload your data from a file for immediate use by your Spark instance. You can set up Object Storage directly from the Spark service interface.

**Pricing Plan:** Monthly Process shown above reflect the: [United States](#)

PLAN	FEATURES	PRICING
<input checked="" type="radio"/> Lite	2 Spark Executors	Free
An entry level plan to run programs using up to 2 Spark executors		
<input type="radio"/> Reserved Enterprise	30 Spark Executors	Expand each section to view details

[Cancel](#) [Create](#)

- Now an Apache Spark service has been added to your Project

## Step 3 - Create a Notebook

- From the top menu, select **Tools**, and then **Notebook**
- Select **From URL**
- Give the notebook a name : **IoT Sensor Analytics**
- In the **Notebook URL** field, paste the following URL:  
<https://raw.githubusercontent.com/binnes/esp8266Workshop/master/part4/notebooks/ESP8266-DHT-IoT-Sensor-Analytics.ipynb>
- Scroll down to the **Select runtime** dropdown and choose your **Spark** runtime

IBM Watson

1421965 - John Walic... JW

My Projects / New Notebook

## New notebook

Blank    From file    From URL

Name\*

IoT Sensor Analytics

30 Characters Remaining

Description

Type your Description here

Notebook URL\*

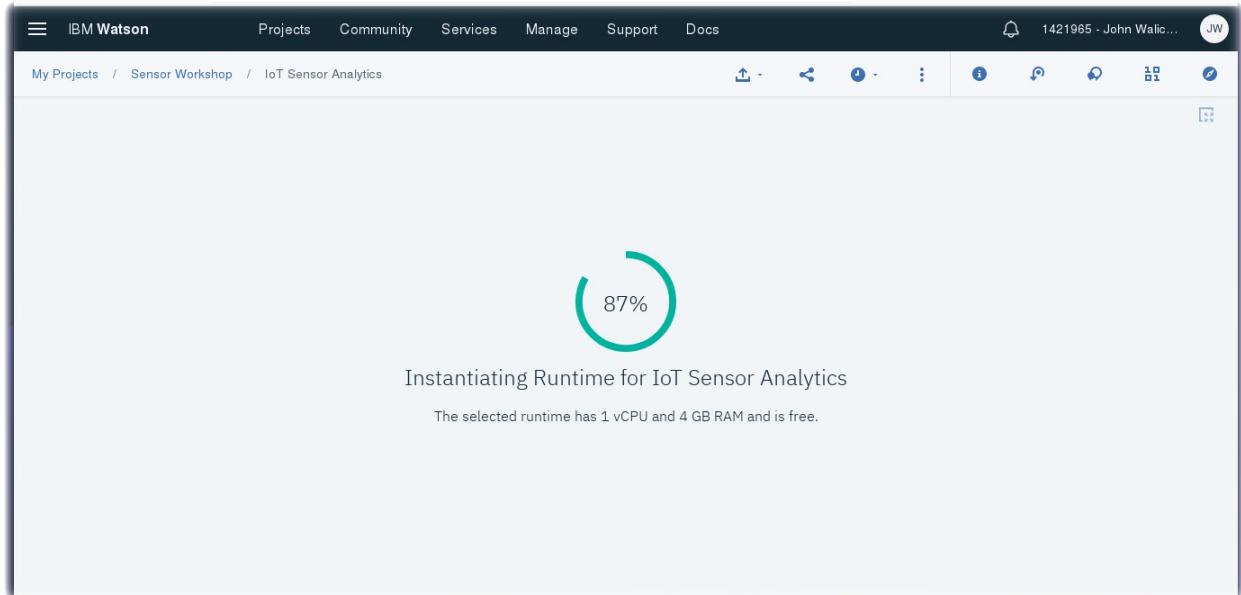
content.com/binnes/esp8266Workshop/master/part4/notebooks/ESP8266-DHT-IoT-Sensor-Analytics.ipynb

Select runtime\* Includes notebook environments ⓘ

spark-zu

Cancel    Create Notebook

- Click on **Create Notebook**



- Before running this notebook, an additional connector notebook needs to be installed.

## Step 4 - Create an Apache Bahir Connector Notebook

---

This will install the Apache Bahir connector within your Project/Apache Spark service. If you restart the kernel or start a new notebook in the same project you can use Apache Bahir for connecting to the Cloudant/Apache CouchDB service.

- From the top menu, select **Tools**, and then **Notebook**
- Select **From URL**
- Give the notebook a name: **Apache Bahir**
- In the **Notebook URL** field, paste the following URL:  
[https://raw.githubusercontent.com/romeokienzler/developerWorks/master/coursera/bahir\\_setup.ipynb](https://raw.githubusercontent.com/romeokienzler/developerWorks/master/coursera/bahir_setup.ipynb)
- Scroll down to the **Select runtime** dropdown and choose your **Spark** runtime
- Click on **Create Notebook**

### Apache Bahir Connector Initialization

- Once the Apache Bahir notebook loads, click on the **Kernel** menu and select **Restart & Run All** button
- Do this **Restart & Run All** step **twice** to complete the package installs

IBM Watson Projects Tools Community Services Manage Support Docs 1421965 - John Walic... JW

My Projects / IoT Sensor Analytics / Apache Bahir

File Edit View Insert Cell Kernel Help Trusted | Python 3.5 with Spark 2.1

**Please run the following cell once.**

This will install the Apache Bahir connector within your Project/Apache Spark service. If you restart the kernel or start a new notebook in the same project you can use Apache Bahir for connecting to the Cloudant/Apache CouchDB service. Please note that this will install a patched version of the connector (since the pull request hasn't been merged with the trunk yet).

You'll find more information on the patch here:

<https://github.com/apache/bahir/pull/49> <https://issues.apache.org/jira/browse/BAHIR-130>

```
In [1]: import pixiedust
pixiedust.installPackage("https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar")
pixiedust.installPackage("com.typesafe:config;1.3.1")
pixiedust.installPackage("com.typesafe.play:play-json_2.11:jar:2.5.9")
pixiedust.installPackage("org.scalaj:scalaj-http_2.11:jar:2.3.0")
pixiedust.installPackage("com.typesafe.play:play-functional_2.11:jar:2.5.9")
```

Pixiedust database opened successfully

 Pixiedust version 1.1.11

```
Downloading package https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar to /gpfs/fs01/user/se24-332237bdbf0d47-0c79bc44c3a6/data/libs/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar
Downloaded 153353 of 153353 bytes
```

```
Package https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar downloaded successfully
Please restart Kernel to complete installation of the new package
Successfully added package https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar
Downloading package com.typesafe:config:1.3.1 to /gpfs/fs01/user/se24-332237bdbf0d47-0c79bc44c3a6/data/libs/config-1.3.1.jar
```

```
Downloaded 153353 of 153353 bytes
```

[https://dataplatform.cloud.ibm.com/datalib/api/v2/notebooks?project\\_id=54e3ec7d-841d-4d31-97f4-0a4544a1381b&api=v2&env=a#](https://dataplatform.cloud.ibm.com/datalib/api/v2/notebooks?project_id=54e3ec7d-841d-4d31-97f4-0a4544a1381b&api=v2&env=a#)

- The packages are successfully installed when the cell reports "Package already installed" messages

IBM Watson Projects Tools Community Services Manage Support Docs 1421965 - John Walic... JW

My Projects / IoT Sensor Analytics / Apache Bahir

File Edit View Insert Cell Kernel Help Trusted | Python 3.5 with Spark 2.1

**Please run the following cell once.**

This will install the Apache Bahir connector within your Project/Apache Spark service. If you restart the kernel or start a new notebook in the same project you can use Apache Bahir for connecting to the Cloudant/Apache CouchDB service. Please note that this will install a patched version of the connector (since the pull request hasn't been merged with the trunk yet).

You'll find more information on the patch here:

<https://github.com/apache/bahir/pull/49> <https://issues.apache.org/jira/browse/BAHIR-130>

```
In [1]: import pixiedust
pixiedust.installPackage("https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar")
pixiedust.installPackage("com.typesafe:config;1.3.1")
pixiedust.installPackage("com.typesafe.play:play-json_2.11:jar:2.5.9")
pixiedust.installPackage("org.scalaj:scalaj-http_2.11:jar:2.3.0")
pixiedust.installPackage("com.typesafe.play:play-functional_2.11:jar:2.5.9")
```

Pixiedust database opened successfully

 Pixiedust version 1.1.11

```
Package already installed: https://github.com/romeokienzler/developerWorks/raw/master/coursera/spark-sql-cloudant_2.11-2.3.0-SNAPSHOT.jar
Package already installed: com.typesafe:config;1.3.1
Package already installed: com.typesafe.play:play-json_2.11:2.5.9
Package already installed: org.scalaj:scalaj-http_2.11:2.3.0
Package already installed: com.typesafe.play:play-functional_2.11:2.5.9
```

```
Out[1]: <pixiedust.packageManager.Package at 0x7fb2d82b1b70>
```

You are now ready to analyse the IoT sensor historical dataset using a Jupyter notebook and Spark. Proceed to the next [Jupyter Notebook section](#).

*Quick links :*

[Home - IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) -

[Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

*Quick links :*

[Home](#) - [IoT Platform Starter](#) - [Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---

# Run a Jupyter Notebook in Watson Studio

---

## Lab Objectives

---

In this lab you will read IoT data into a Watson Studio Project Jupyter Notebook and perform some analytics. You will learn:

- Jupyter Notebooks
- Read data from a Cloudant DB into Spark
- Calculate the Average, Standard Deviation, and Find the Min/Max

## Introduction

Jupyter Notebook is a web-based interactive computational environment for interactive data science and scientific computing. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension. A Jupyter kernel is a program responsible for handling various types of request (code execution, code completions, inspection), and providing a reply.

## Step 1 - Cloudant Credentials

Before we can read the ST Microelectronics Discovery Kit IoT Node temperature and humidity data into a Jupyter notebook we need to create credentials to the Cloudant database created in the [Store Data in Cloud Storage](#) section.

- Open a new browser tab.
- Return to the [IBM Cloud dashboard](#) and your IoT Starter application. Click on the cloudantNoSQLDB service connection (1).

The screenshot shows the IBM Cloud application overview page for a Node.js application named 'stmicro-discovery-kit-iot-node'. The left sidebar includes links for Getting started, Overview, Runtime, Connections, Logs, Monitoring, and API Management. The main area displays runtime details: Buildpack (.js), Instances (1), MB Memory per Instance (256), and Total MB Allocation (256). Below this, the 'Connections' section lists two entries: 'stmicro-discovery-kit-iot-node-cloudantNoSQL...' and 'stmicro-discovery-kit-iot-node-iotf-service'. A red circle labeled '1' points to the first connection entry, and a red box highlights the 'stmicro-discovery-kit-iot-node-cloudantNoSQL...' entry.

- Read about the Cloudant Storage service and click on the **Service credentials** menu item in the left menu bar.

The screenshot shows the IBM Cloud service credentials page for the 'stmicro-discovery-kit-iot-node-cloudantNoSQLDB' service. The left sidebar has 'Connections' and 'Service credentials' selected. The main area shows the service name, location (US South), organization (walicki@us.ibm.com), and space (dev). It also shows an alias 'Alias of Cloudant-Id' with a red box around it. Below this, the 'Connected Applications' section lists one application: 'stmicro-discovery-kit-iot-node' with route 'stmicro-discovery-kit-iot-node.mybluemix.net' and status 'Running'.

- Click on **New credential**

The screenshot shows the IBM Cloud interface for managing service credentials. The left sidebar has 'Service credentials' selected. The main area displays the 'esp8266-workshop-cloudantNoSQLDB' instance details, including location (US South), organization (Org: walicki@us.ibm.com), and space (Space: team). A 'Service credentials' section contains a JSON snippet for Cloudant credentials. A 'New credential' button is available to add more. A note at the bottom says 'Click New credentials to create a set of credentials for this instance'.

- Give your credential a name: **Credentials-DSX**
- Click on **Add**
- Expand the **View credentials** twistie
- The Cloudant hostname, user and password credentials will be displayed.
- Keep this browser tab open. You will use these credentials in the next Step.

## Step 2 - Run a Jupyter Notebook

- Return to the Watson Studio browser tab and open the **STM Discovery Kit IoT Sensor Analytics** notebook.

The screenshot shows the Watson Studio interface with the 'Notebooks' section open. It lists two notebooks: 'ApacheBahir' and 'ESP8266-DHT-IoT-Sensor-Analytics'. Both are in Python 3.5, created by John Walicki on April 18, 2018. The second notebook is currently selected.

NAME	SHARED	SCHEDULED	STATUS	LANGUAGE	LAST EDITOR	LAST MODIFIED	ACTIONS
ApacheBahir			○	Python 3.5	John Walicki	18 Apr 2018	
ESP8266-DHT-IoT-Sensor-Analytics			○	Python 3.5	John Walicki	18 Apr 2018	

- Make certain you are in **Edit** mode by clicking on the Pencil icon.
- You will now copy the Cloudant hostname, user and password credentials to your **STM Discovery Kit IoT Sensor Analytics** notebook cell 5

The screenshot shows a Jupyter Notebook interface within the IBM Watson environment. The top navigation bar includes 'IBM Watson', 'Projects', 'Tools', 'Community', 'Services', 'Docs', 'Support', 'Manage', and a user icon. The main area displays a series of code cells:

```

In [ ]: # To confirm you have installed the latest version of PixieDust on your system, run this cell
!pip install --ignore-installed --upgrade pixiedust

In [ ]: import pixiedust

In [ ]: !pip list --format=columns

In [ ]: def readDataFrameFromCloudant(database):
    cloudantdata=spark.read.load(database, "org.apache.bahir.cloudant")
    cloudantdata.createOrReplaceTempView("historicaldata1")
    spark.sql("SELECT * from historicaldata1").show()
    return cloudantdata

In [ ]: # Credentials from your cloudantNoSQLDB service
hostname = <your cloudant hostname credential here -bluemix.cloudant.com>
user = <your cloudant user credential here -bluemix>
pw = <Your cloudant password credential here>

database = "historicaldata1" #as long as you didn't change this in the NodeRED flow the database name stays the same

In [ ]: spark = SparkSession \
    .builder \
    .appName("Cloudant Spark SQL Example in Python using temp tables") \
    .config("cloudant.host",hostname) \
    .config("cloudant.username", user) \
    .config("cloudant.password", pw) \
    .getOrCreate()

cloudantdata=readDataFrameFromCloudant(database)

In [ ]: display(cloudantdata)

In [ ]: df = spark.sql("select avg(temp), std(temp), max(temp), min(temp) from historicaldata1")
df.show()
display(df)

```

A blue message icon is visible in the bottom right corner of the code area.

- Move the focus to the first cell.
- In the toolbar, click on the **Run** button to run each cell.
- Spark will calculate the Average, Standard Deviation, Max and Min of your DiscoveryKit IoT Historical data set.

---

#### *Quick links :*

[Home - IoT Platform Starter - Device Types and Devices](#) - [Node-RED Setup](#) - [Sensor Data](#) - [Node-RED Charts](#) - [Store Data in Cloud Storage](#) - [Historical Charts](#) - [Watson Studio](#) - [Jupyter Notebooks](#)

---