InterConnect 2016

The Premier Cloud & Mobile Conference

Session: IDA-6158

Connect, Capture, Analyze, Decide: Get Hands-On with Sensors and Watson IoT Platform

Lab Instructions

Author

Bryan Boyd, Watson IoT

bboyd@us.ibm.com

February 21 – 25

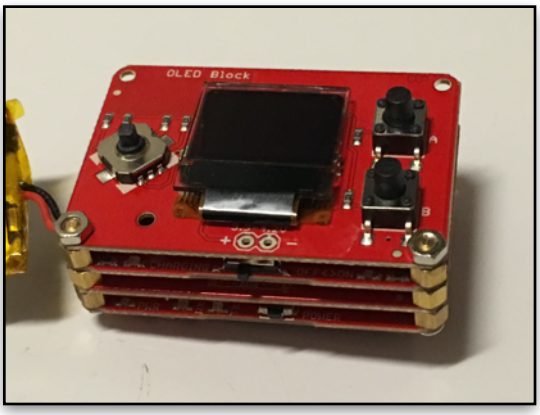
MGM Grand & Mandalay Bay

Las Vegas, Nevada

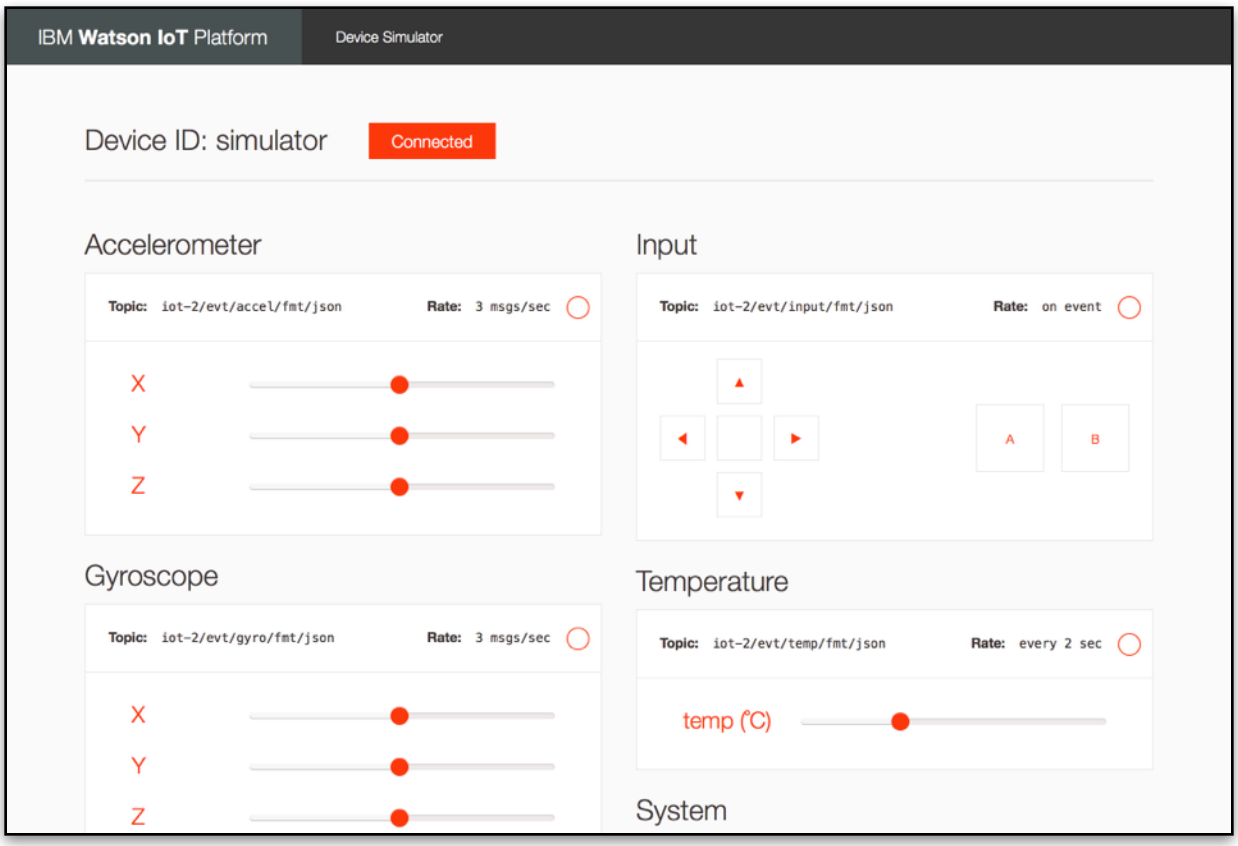
Overview

This lab allows you to interact with real-time data from devices connected to Watson IoT Platform. In this lab you will register and connect a simulated device to a Watson IoT Platform organization, view data and build real-time charts in the platform dashboard. You will then run an analytics application to detect and trigger alerts based on the incoming real-time event data.

The device used in this lab is an **Intel Edison** with SparkFun Blocks for sensor data and input. The Edison connects to Watson IoT Platform and publishes event data using the real-time MQTT API.



Rather than using physical sensors for the lab, you will run a web-based simulator that models the behavior of the device in browser.



Section 1 - Setup

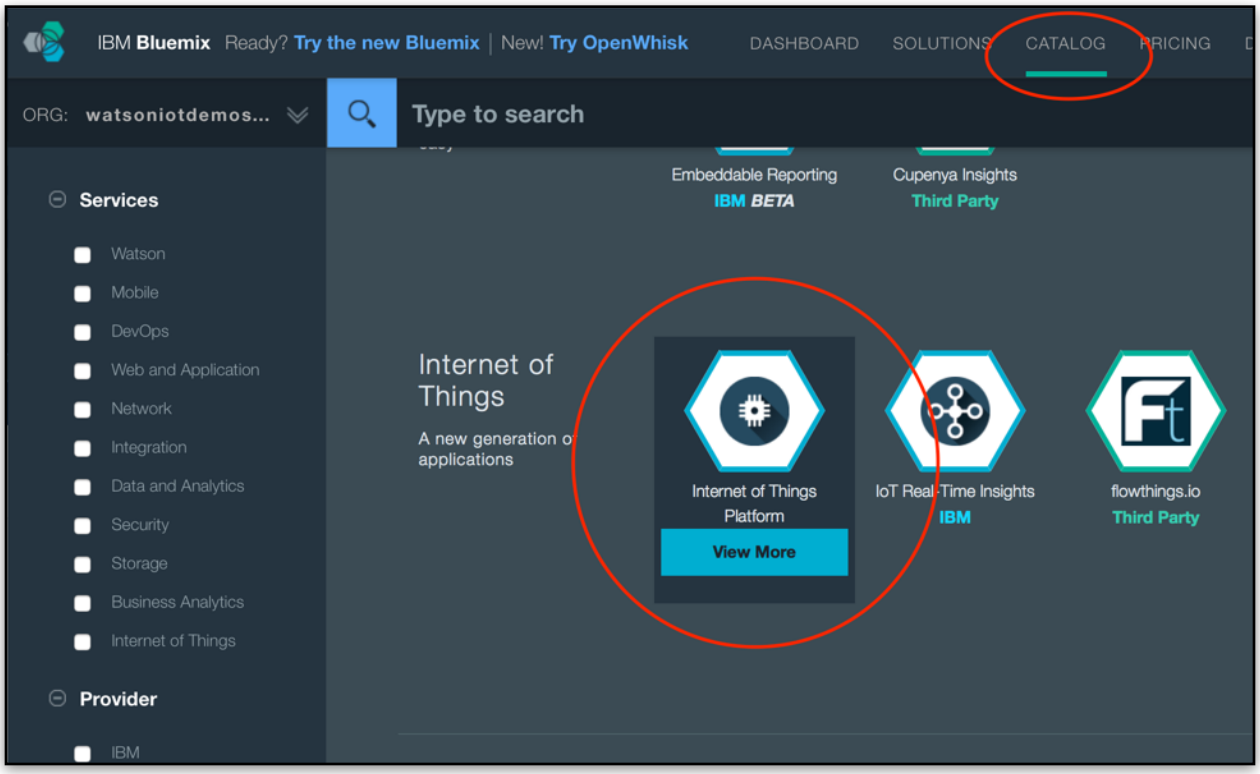
Sign up for Bluemix

This lab will require a Bluemix account. If you do not have access to a Bluemix account, you can register for a free 30-day trial at the following link:

https://console.ng.bluemix.net/registration

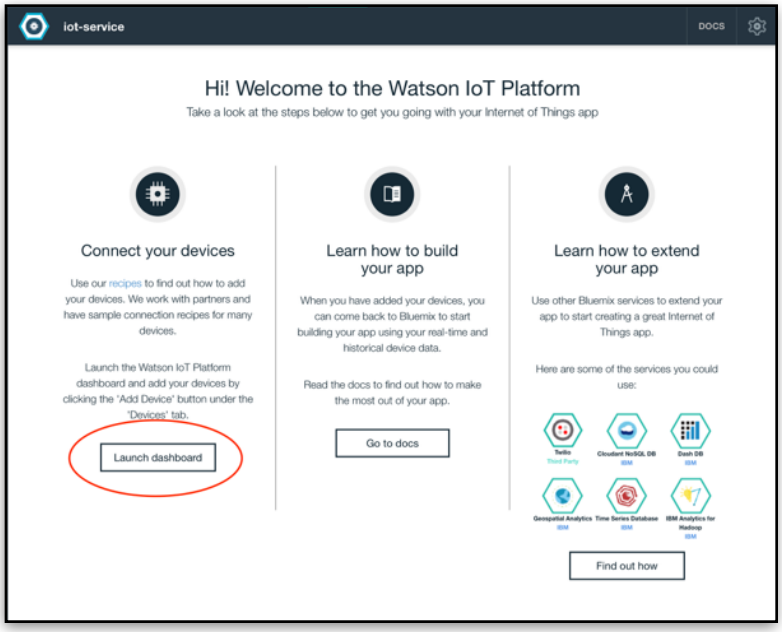
Create a Watson IoT Platform organization

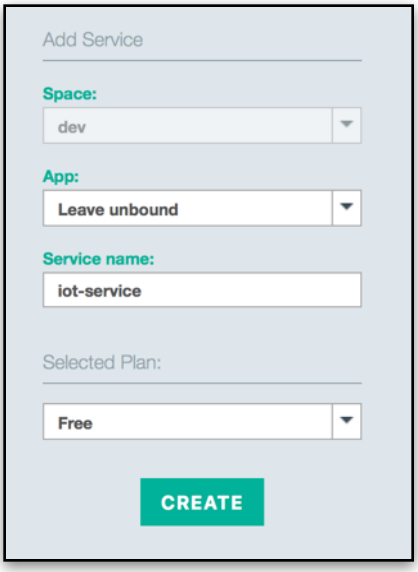
From your Bluemix account, create a new **Internet of Things Platform** service.



Select the **Free** plan, leave the service **unbound** and give the service a name you will remember (example: **iot-service**). Press **Create** to create aWatson IoT Platform organization.

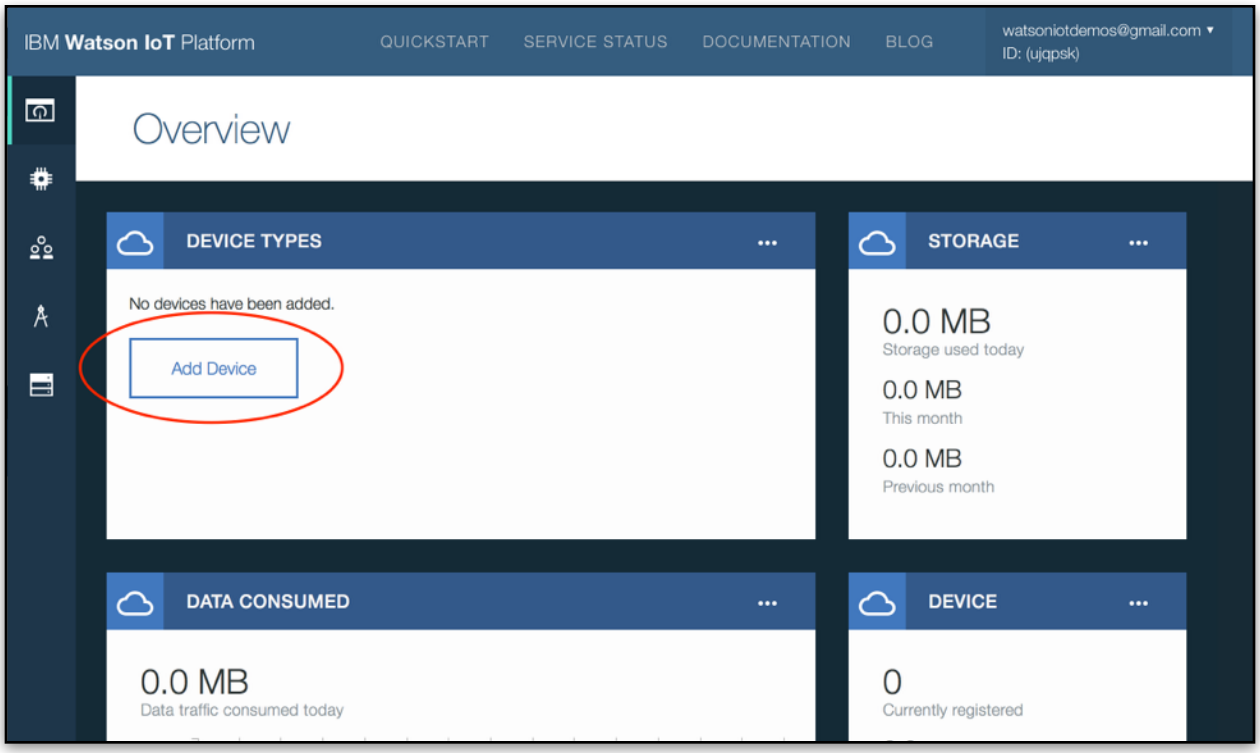
On the following screen, select **Launch Dashboard** to enter your organization dashboard.



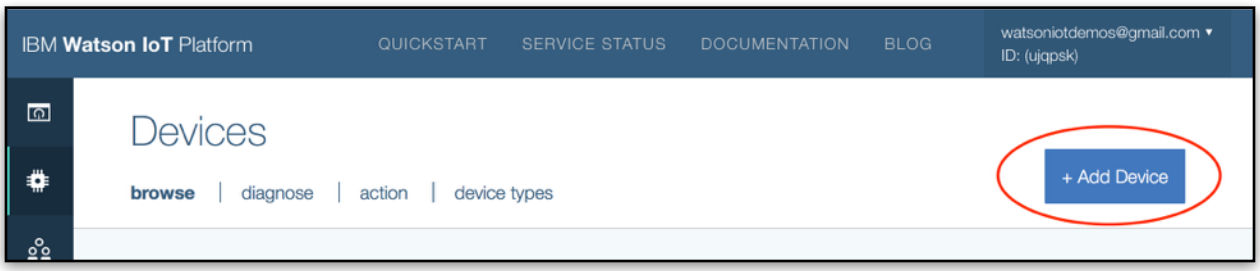


Add a device

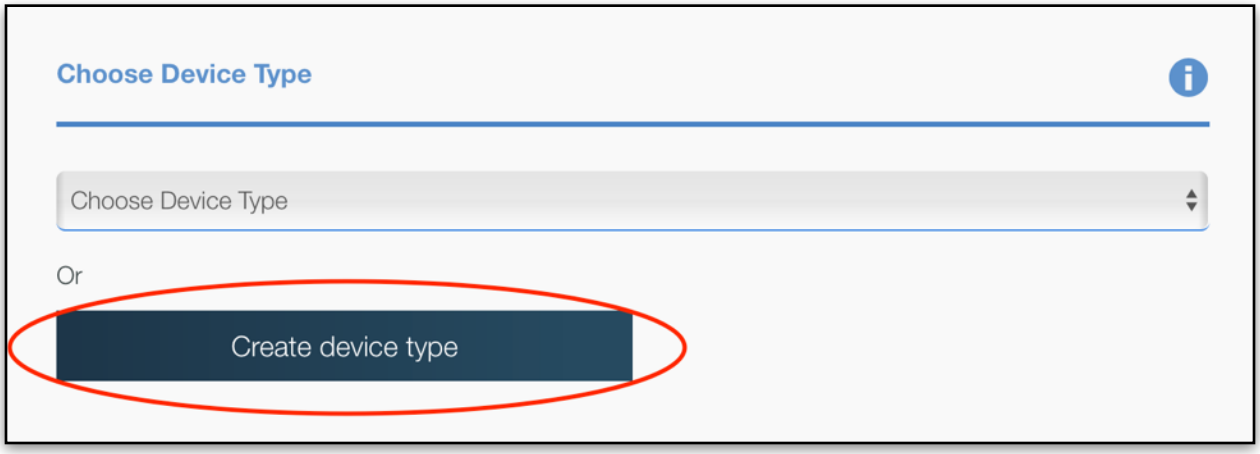
Before connecting the simulator, you first have to add the device to your organization from the dashboard. If there are no devices in your organization, this process can be started from the **Add Device** button in the Overview page.

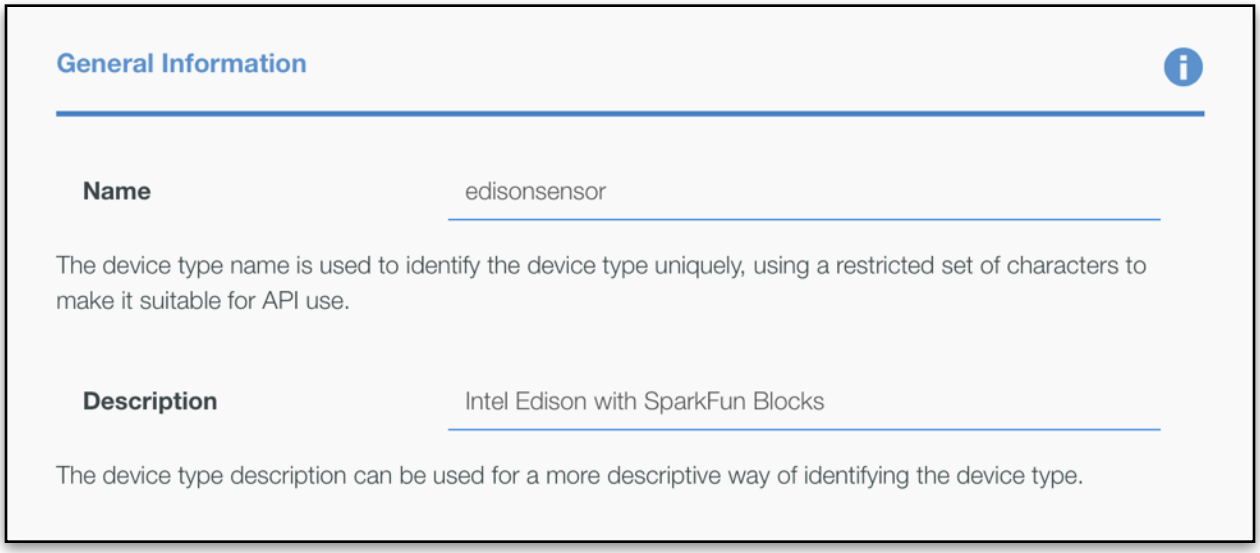


Alternatively, you can add a device from the Devices page in the dashboard.



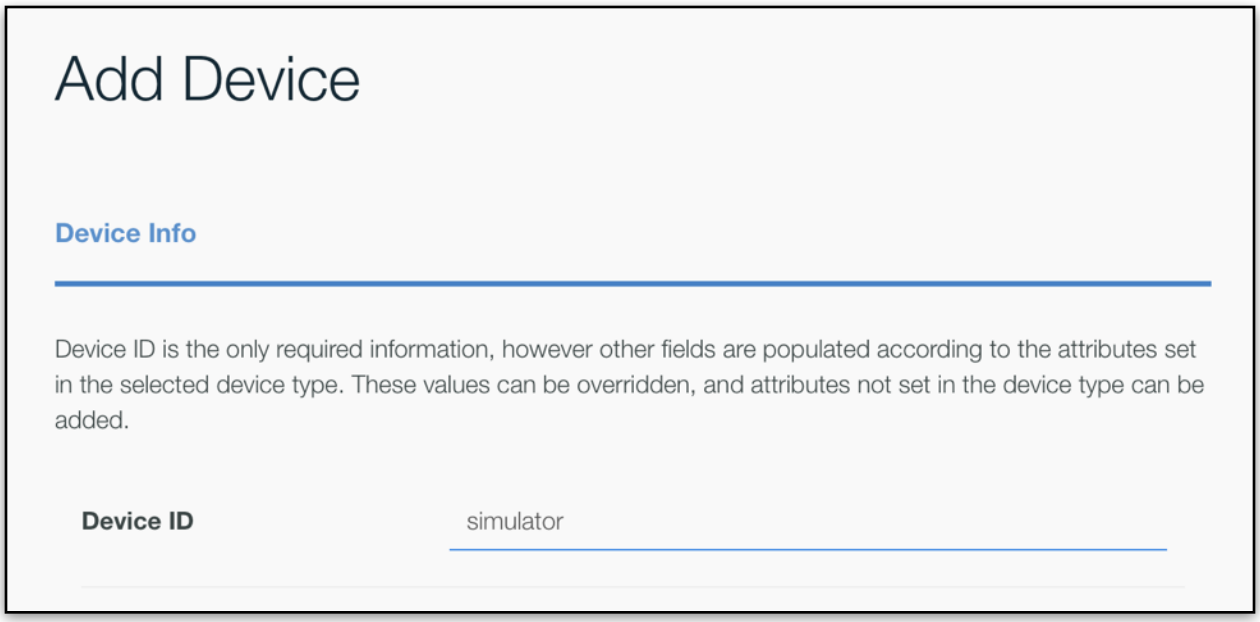
A device must be associated with a **device type**. First, create a device type named **edisonsensor** and provide a description. You do not need to specify a template for the device type, or add metadata.





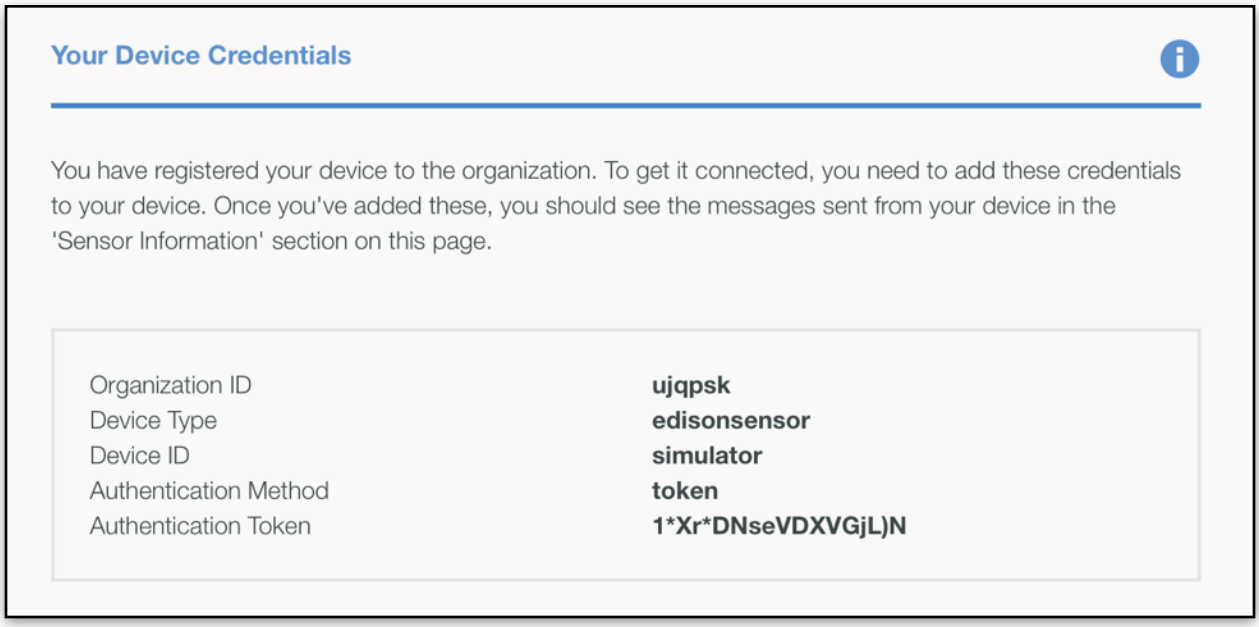
Once the device type is created, continue the process of adding the device by specifying a **Device ID**.

NOTE: The rest of this tutorial will assume the ID **simulator** is used for the device. If you choose to use a different ID, enter your ID where **simulator** is referenced.



Continue the Add Device process. When prompted to provide a token, leave the field blank. The Watson IoT Platform will generate a unique token for your device.

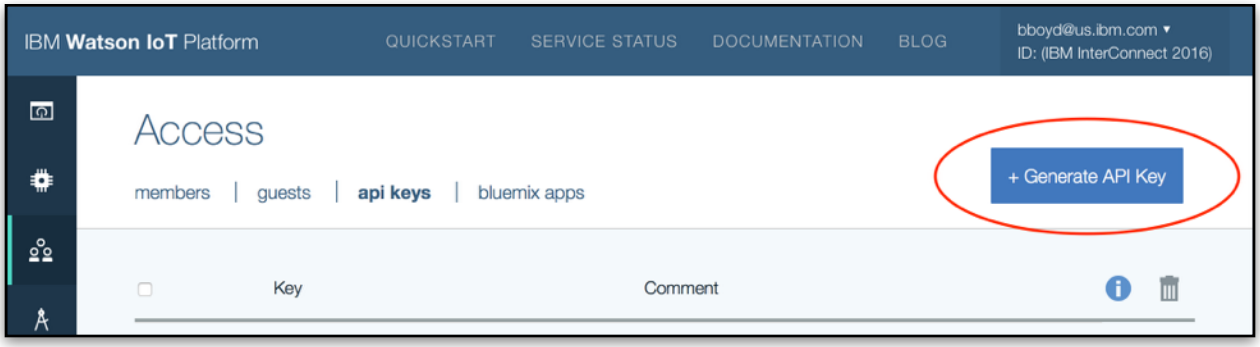
Once the device is added, take note of the organization ID, device type, device ID, and authentication token. You will need to enter these values in the device simulator configuration.

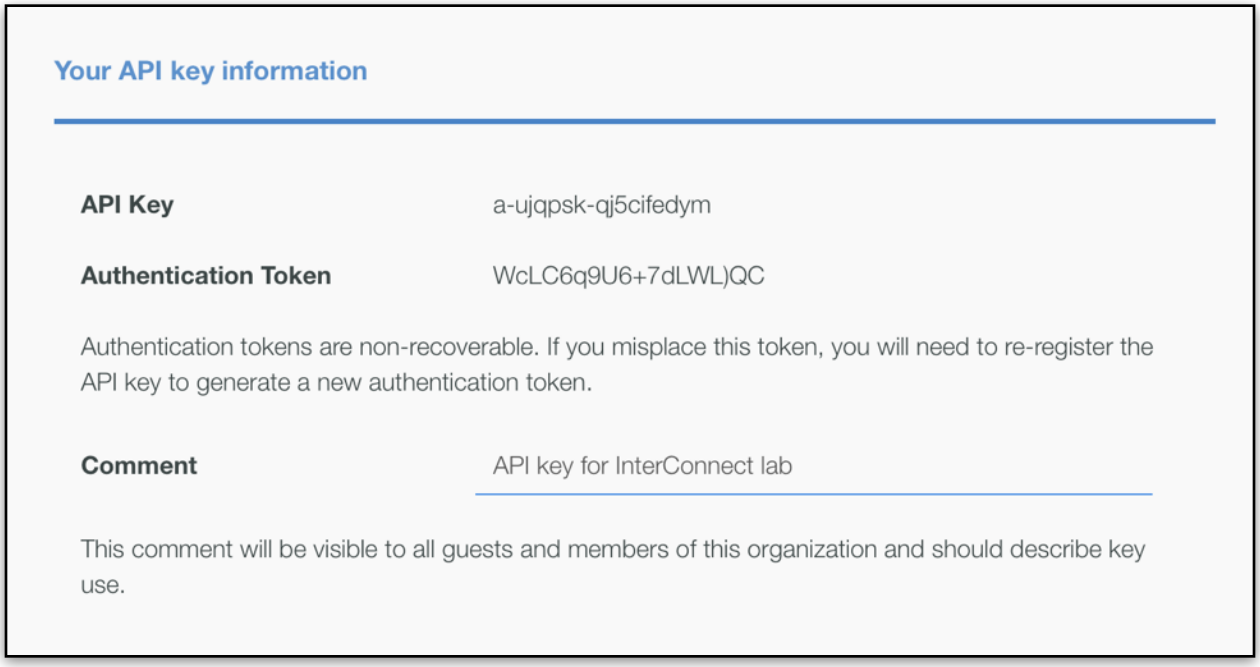


Generate an API key

After adding the device you will need to generate an API key. An API key allows one or more **applications** to connect to Watson IoT Platform and use the MQTT and HTTP APIs of the platform. In this tutorial, the API key will be used by the analytics application to subscribe to device event data, and publish device alert events.

Navigate to the **api keys** tab of the Access page and press **Generate API Key**





Copy down the API key and API token. You will use these credentials to configure the simulator and analytics application.

Section 2 - Connect Sensor

Download the simulator

The next step after adding the device to your organization and generating an API key is to download, configure, and run the device simulator.

Open the following link in a browser in order to obtain the simulator code:
https://ibm.box.com/IDA-6158

You will be prompted for a password: **interconnect2016**

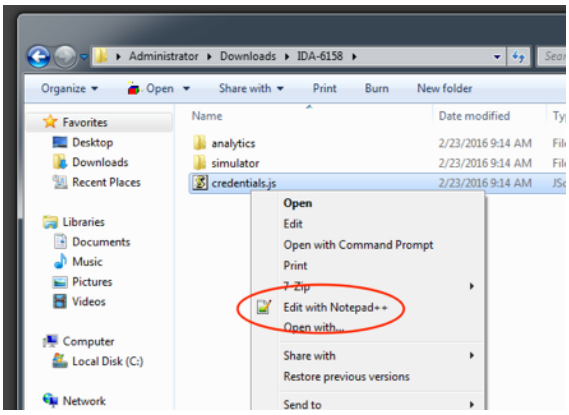
After downloading the file, unzip it on your machine.

NOTE: for the remainder of this document the root location of this code will be referred to as **IDA-6158/**

The zip file of lab material has two folders — **analytics/** and **simulator/** — and a file (**credentials.js**) that both applications reference when connecting to Watson IoT Platform.

Configure the simulator

Open **IDA-6158/credentials.js** in a text editor (example: **Notepad++**). In Windows, this can be done by right-clicking on the file in Explorer and selecting **Edit with Notepad++**.



Replace **org, apiKey, apiToken,** and **authToken** with the appropriate values you recorded from the previous steps.

For example:

```
var Credentials = {
      org: "ujqpsk",
      apiKey: "a-ujqpsk-qj5cifedym",
```

```
      apiToken: "WcLC6q9U6+7dLWL)QC",
      typeId: "edisonsensor",
      deviceId: "simulator",
      authToken: "1*Xr*DNseVDXVGjL)N"
}

module.exports = Credentials;
```

Save the file.

Run the simulator

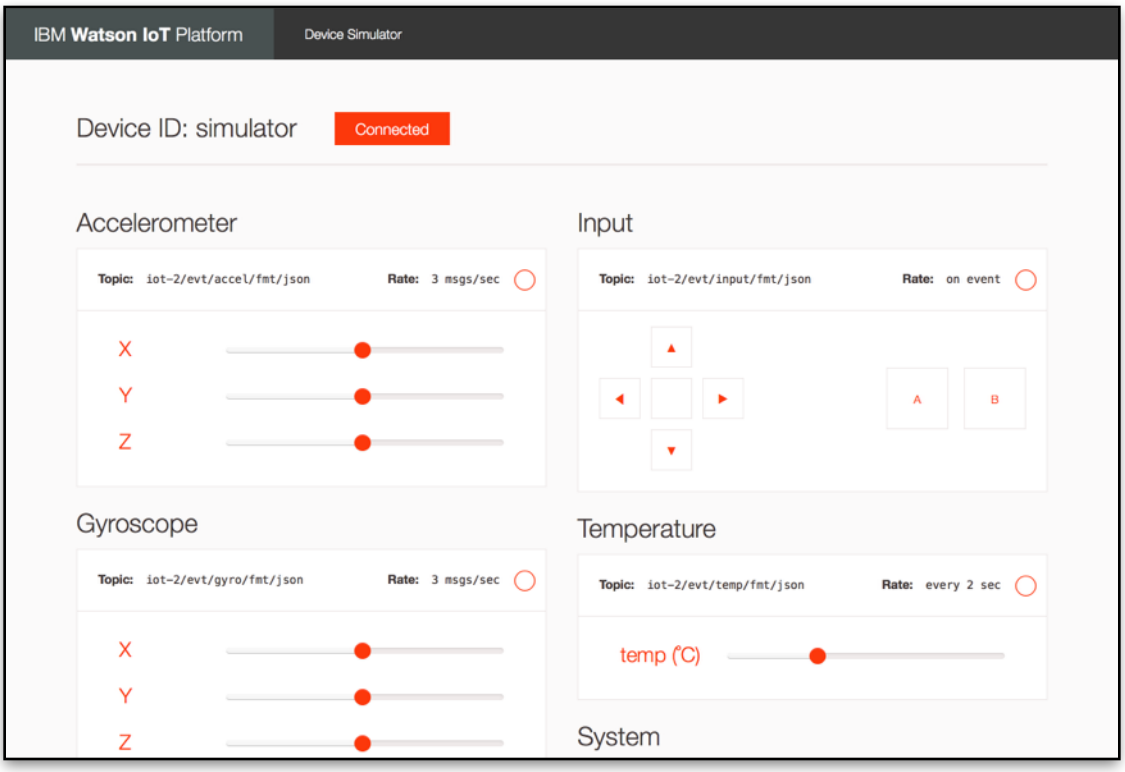
The simulator webapp is run from a Node.js web server. To run the server, perform the following steps:

1) Open a command prompt and navigate to the **IDA-6158/simulator/** folder.
2) Install the Node.js modules required to run the web server: **npm install**
3) Run the Node.js application: **node app.js**
4) If a Windows Firewall warning appears, press **Allow access**.
5) The command prompt will read “server starting on localhost:PORT” (PORT will vary)
6) Open a browser (example: Google Chrome) and navigate to this address.

If your configuration was correct, you will see the device simulator load and connect in the web browser. The web application loads the **credentials.js** file previously edited and establishes an MQTT client connection to Watson IoT Platform using the credentials of the registered device from the Add Device step.

The simulator models a diverse set of device properties that is representative of the complex types of devices that customers connect into the Watson IoT Platform. The simulator groups data into a number of different **events** that applications can subscribe to independently. The events (with an example JSON payload) are as follows:

accel = { d: { x: 0.1, y: -0.3, z: 0.7 } }
gyro = { d: { x: 147, y: 129, z: -242 } }
mag = { d: { x: 0.0, y: -0.1, z: -1.3 } }
input = { d: { UP: false, DOWN: false, LEFT: true, RIGHT: false, SELECT: false, A: false, B: false } }
temp = { d: { temp: 18.6 } }
system = { d: { cpuLoadAvg: 0.27, freeMemory: 293849719, wlan0: “192.168.1.243”, location: { latitude: 36.105531, longitude: -115.181123 } } }



The accel, gyro, and mag events come from a SparkFun 9DOF block and are sampled and published to Watson IoT Platform every 0.5 seconds. The input event is published when an Input button is clicked (or in the case of the real device, a physical joystick/button is pressed).

The system event combines CPU/memory data from the device with the results of an API call to retrieve the approximate location based on the device's IP address.

View device events

The device events can be seen in the Device Drilldown view in the Watson IoT Platform organization dashboard. Navigate to the Devices —> Browse page and select your device from the table. The dashboard UI will parse incoming device events and list them as Event / Datapoint pairs in the **Sensor Information** section of the drilldown page.

Open the simulator and dashboard pages side-by-side, and interact with the simulator (move sliders, press buttons): you should see the sensor information change accordingly in the dashboard.

Sensor Information

Event	Datapoint	Value	Time Received
temp	**d.temp**	**5.1**	**Feb 23, 2016 9:43:41 AM**
mag	**d.x**	**1.32**	**Feb 23, 2016 9:43:41 AM**
mag	**d.y**	**1.83**	**Feb 23, 2016 9:43:41 AM**
mag	**d.z**	**0.84**	**Feb 23,**

Section 3 - Visualize Data

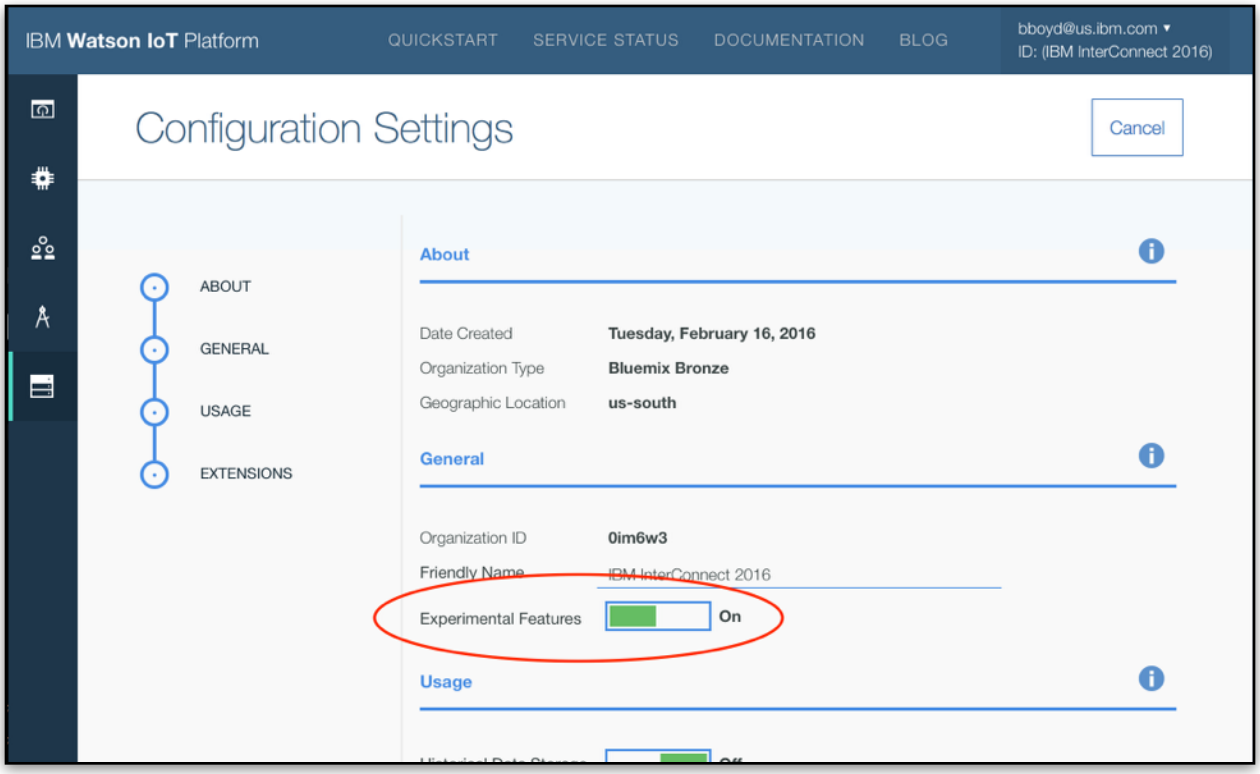
Next, we will use the data visualization cards in Watson IoT Platform to visualize our device data from the organization dashboard.

NOTE: At the time of this writing, the data visualization cards are classified as an experimental feature of the platform and subject to change. The process for adding and configuring cards may be different from what is described in this section.

Experimental features must be enabled on each browser session, and cards created in this section are stored in the browser's local storage.

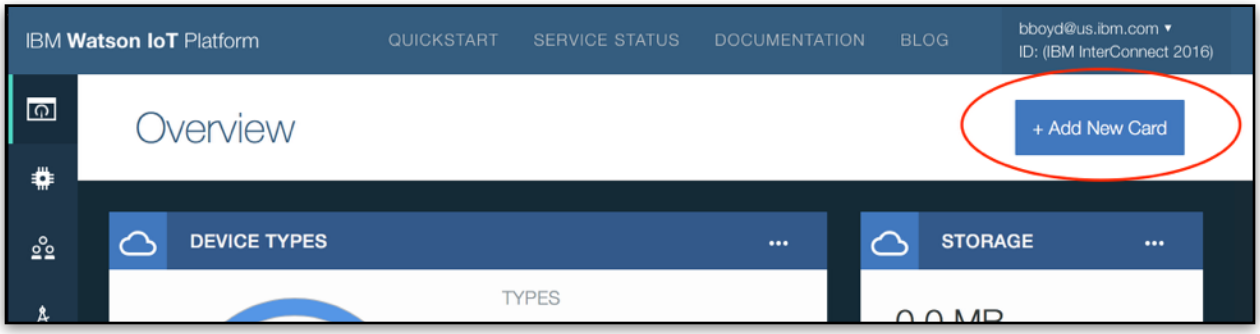
Enable experimental features

Navigate to the Settings page of the Watson IoT Platform organization dashboard. Toggle **Experimental Features** on, and press **Confirm all changes** at the bottom of the page.

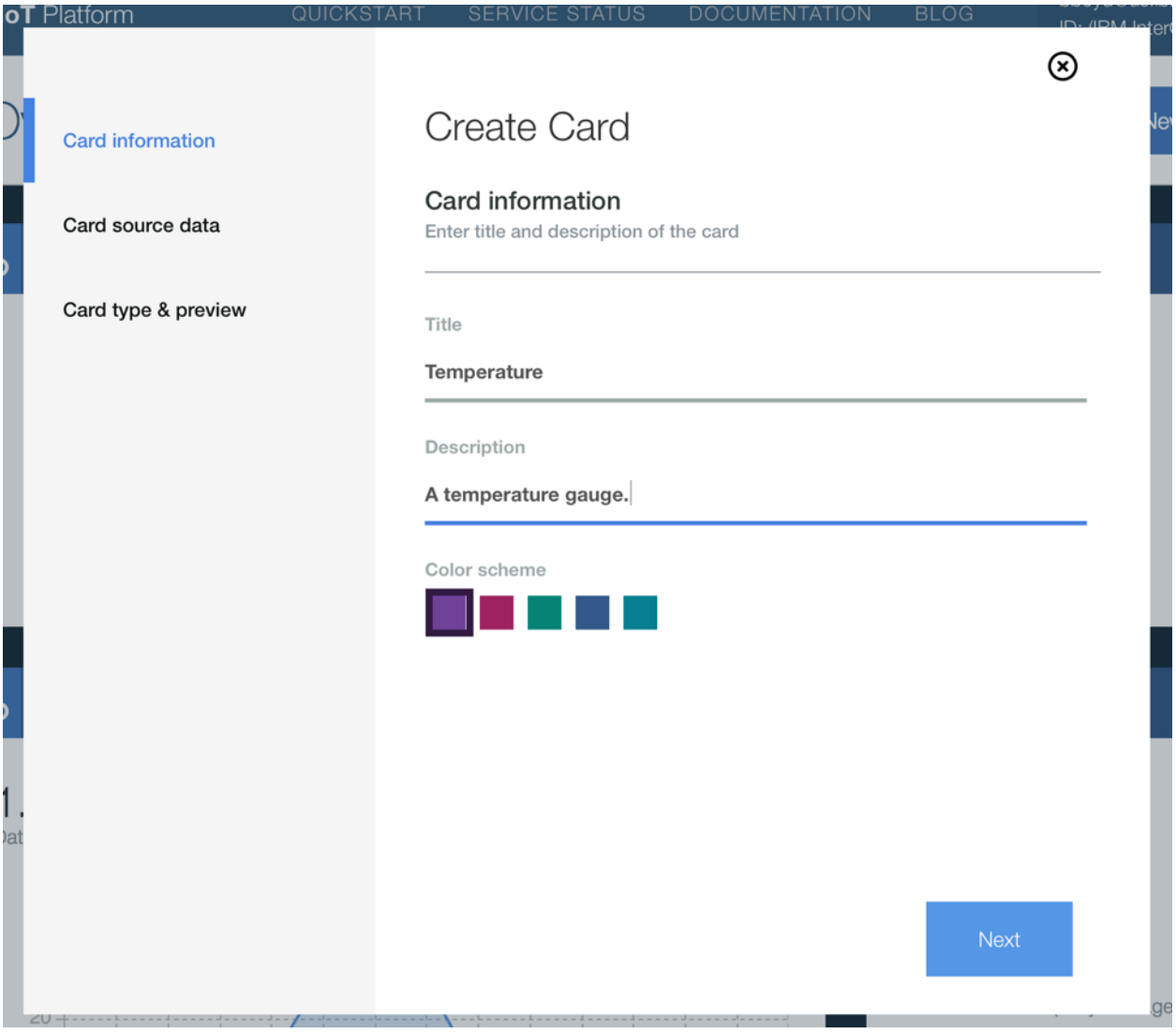


Card 1: Temperature Gauge

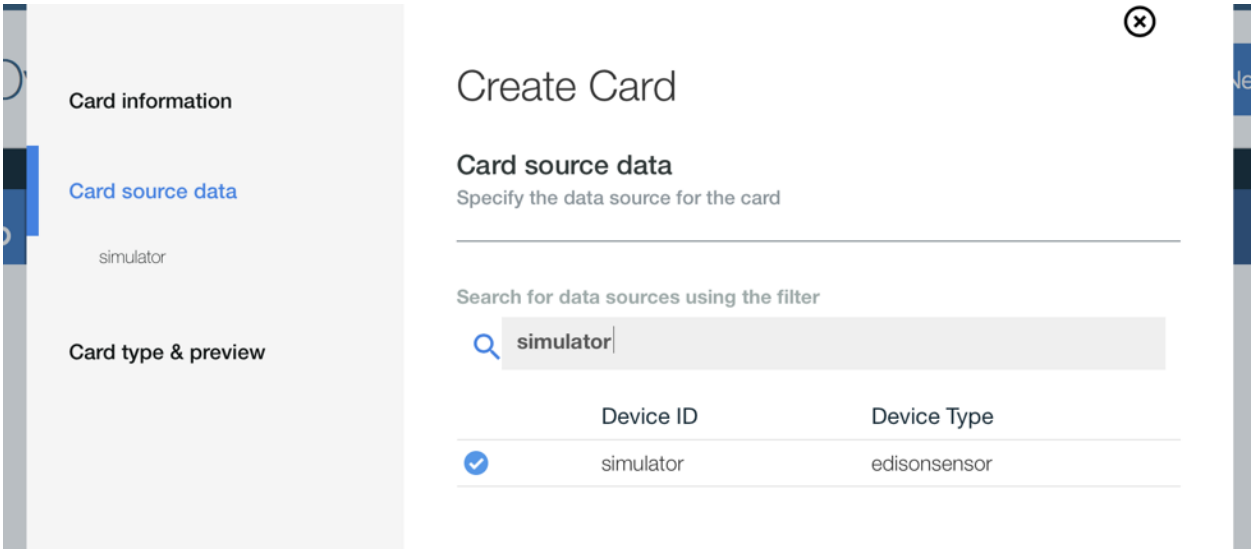
The first card you will create is a gauge to monitor the temperature sensor value from the device. On the Overview page, with experimental features enabled you will now see a button to add a new card.

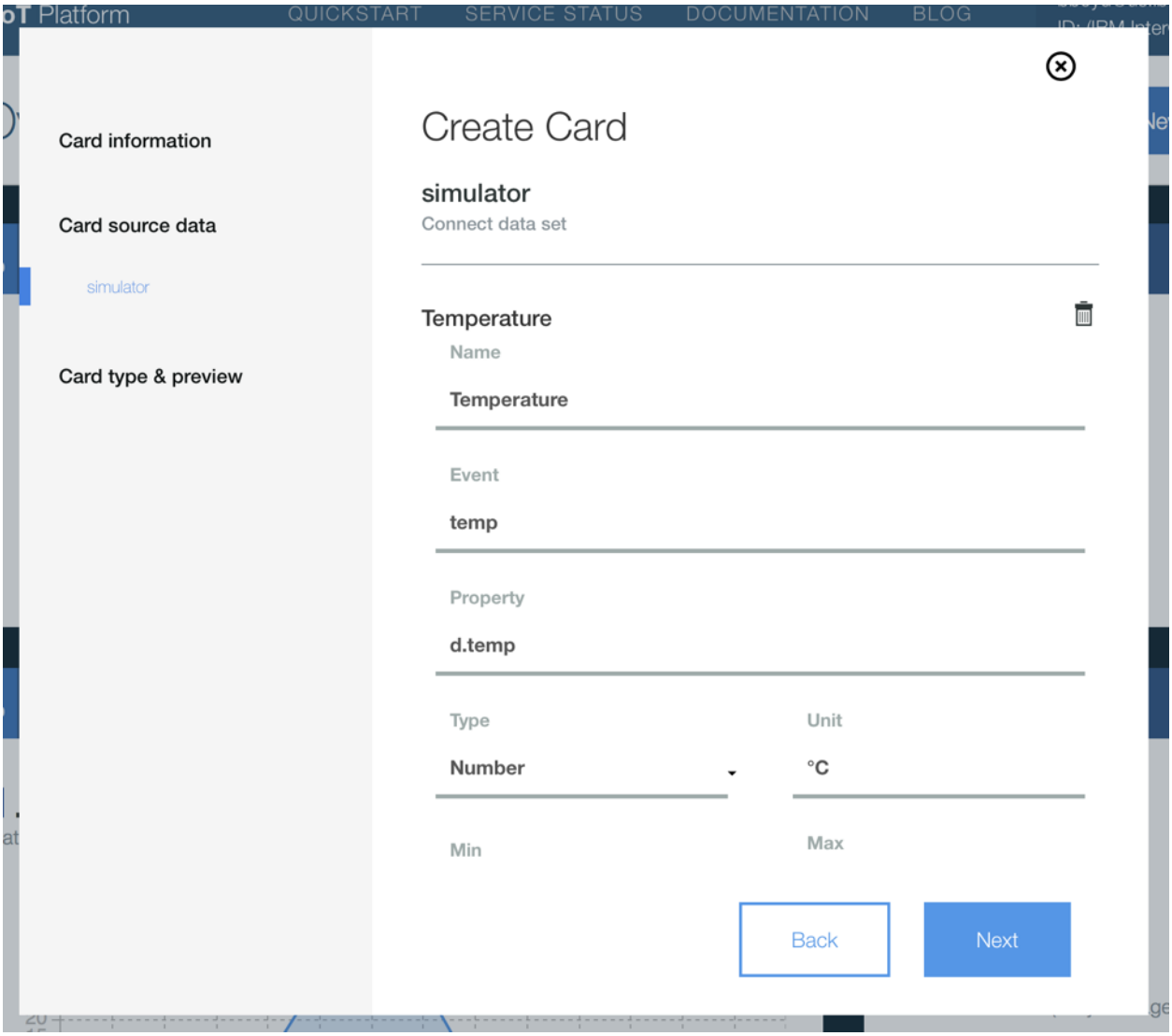


Create a new card, and provide a title, description, and color. The **title** and **color** fields will affect the visual appearance of the created card.



Next, you will specify the **card data source**: the device(s) that will be visualized in the card. Select the **simulator** device you previously added and press **Next**.

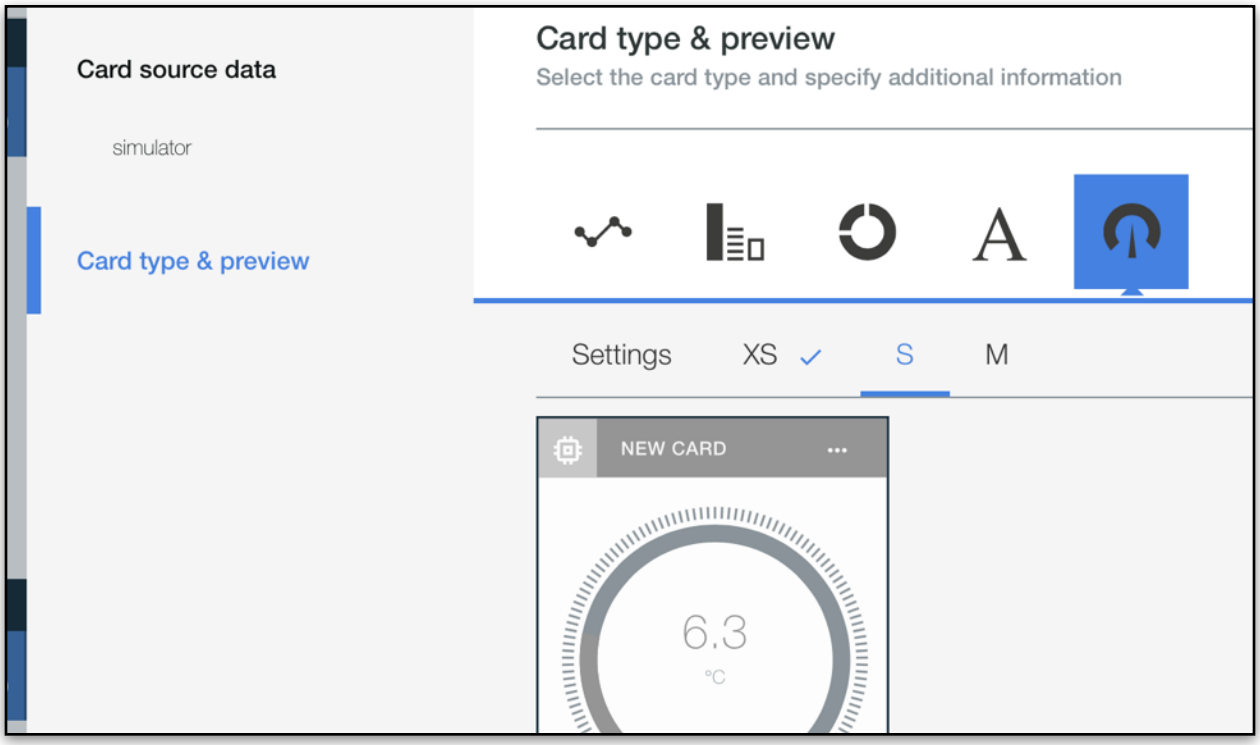




You will now specify one or more **data sets** for each data source. A data set corresponds to one specific sensor value from a device event JSON payload. Create a data set with the name **Temperature**, the event **temp**, property **d.temp**, type **Number**, unit **˚C**, min **0**, max **30**.

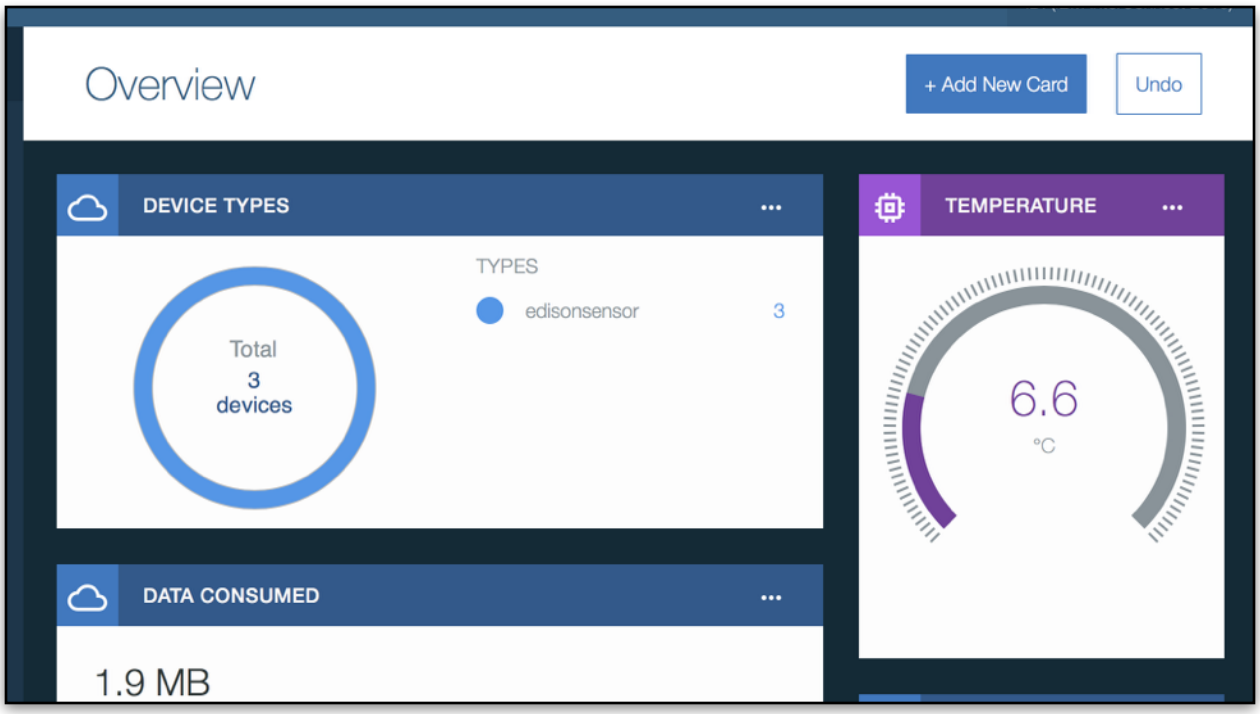
NOTE: If the device is actively publishing events to Watson IoT Platform, these events and properties will appear as suggestions in the input form.

In the final step, you can select and customize the card type. Select the Gauge icon and the Small size, then press **Submit** to create the card.



NOTE: The card settings allow you to customize ranges for the gauge card and set alert thresholds. You can try to customize the card further, or continue with the lab.

The card can be dragged, resized, and edited after it is created. Explore these options using the menu at the top of the icon.



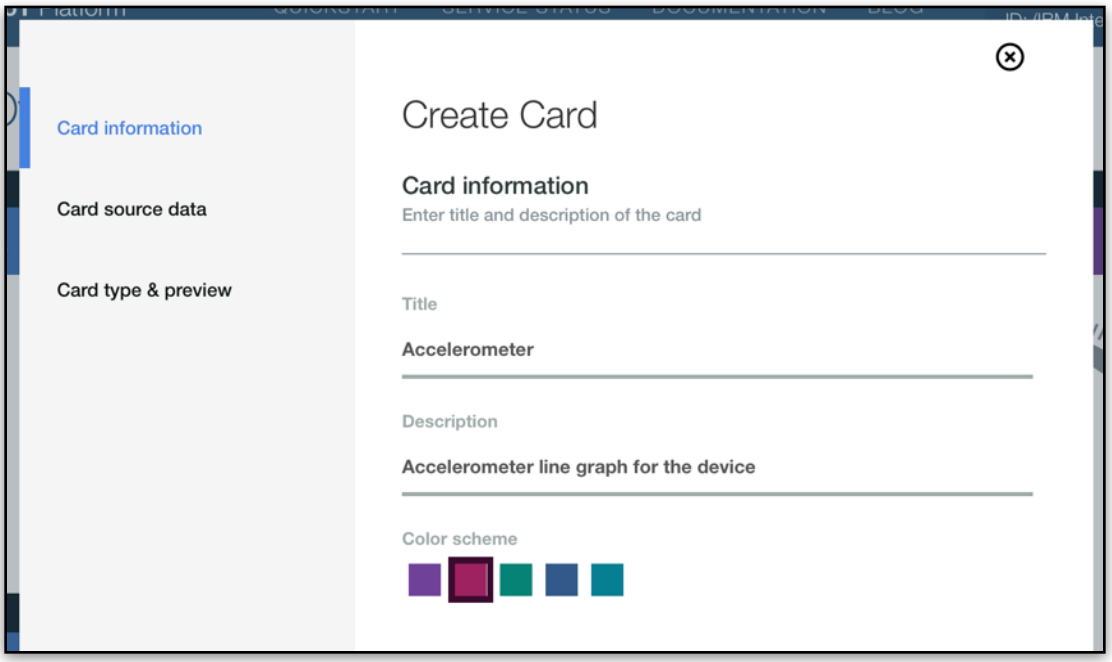
Card 2: Accelerometer Line Chart

You will next create a line chart that models the real-time accelerometer data from the device. This chart will display 3 lines, one for each axis of accelerometer data (X, Y, Z).

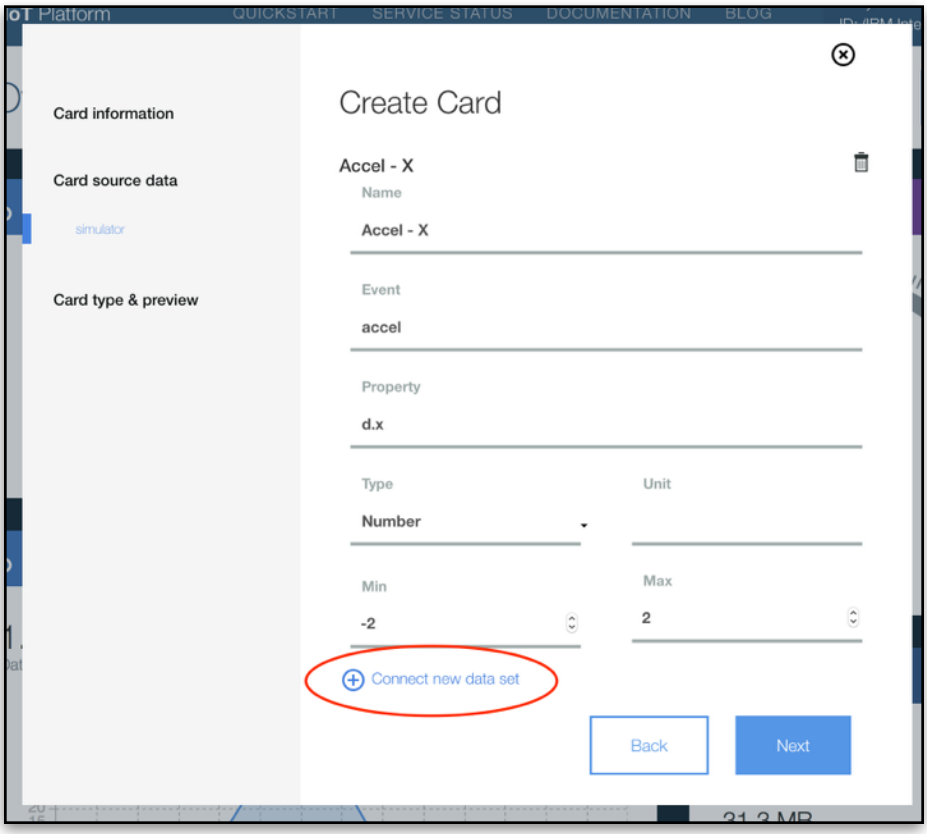
Follow the steps in the previous section to add a new card with the title “Accelerometer”, and a data source of the simulator device.

However, with this card you will configure three data sets for the data source — each having a property that corresponds to the accelerometer axis data.

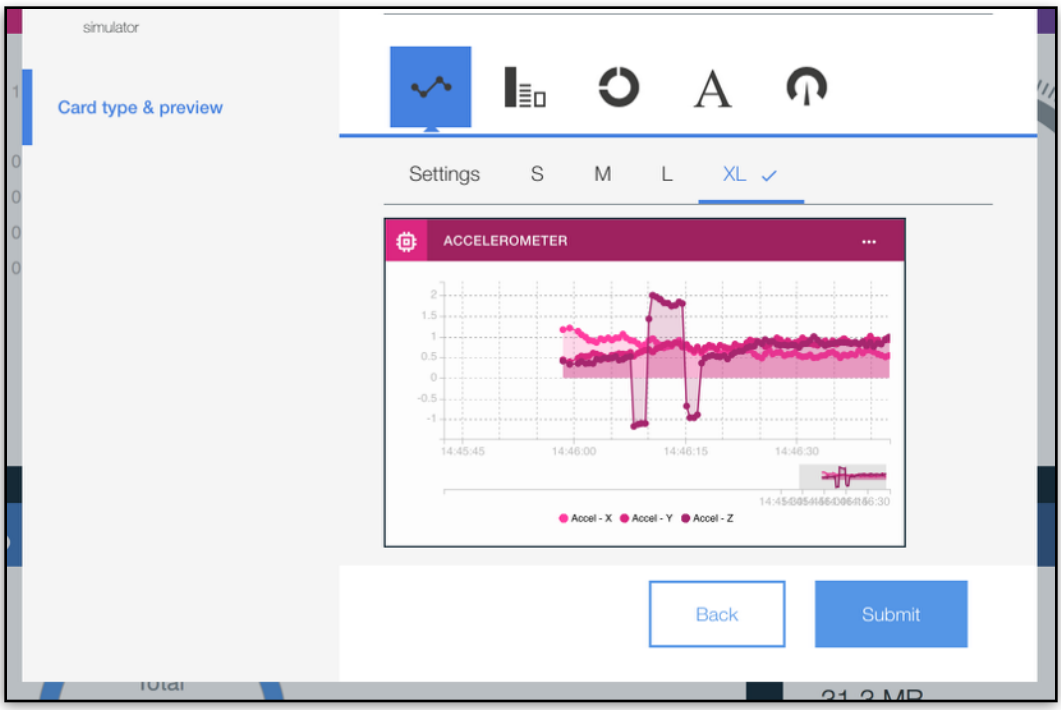
Give the first data set the name **Accel - X**, with event **accel**, property **d.x**, type **Number**, and min/max **-2 / 2**.



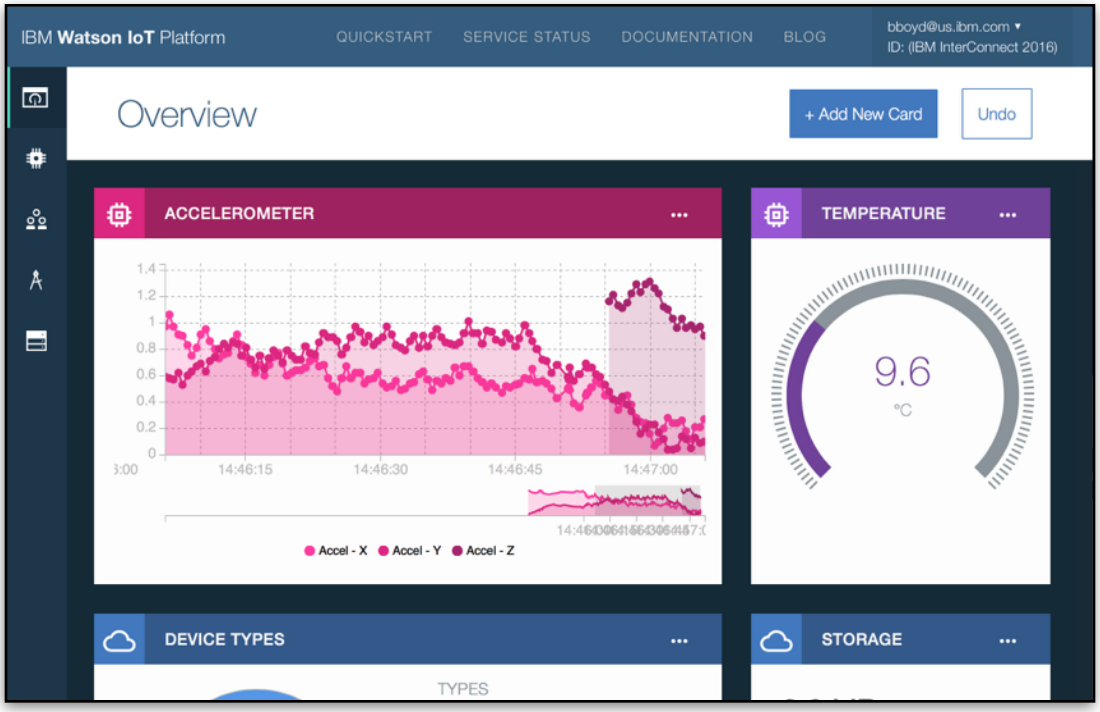
After creating a data set for **Accel - Z**, press **Connect new data set** to add an additional data set (Accel - Y) for the same data source. Repeat for Accel - Z



When selecting the card type, choose the Line Graph option, size **XL**. If the simulator is actively publishing data, you will see a live preview of the chart.



Press **Submit** to create the card.

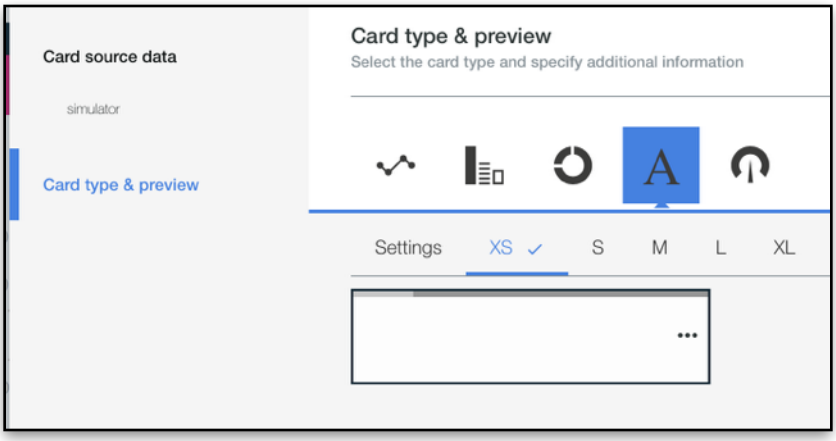


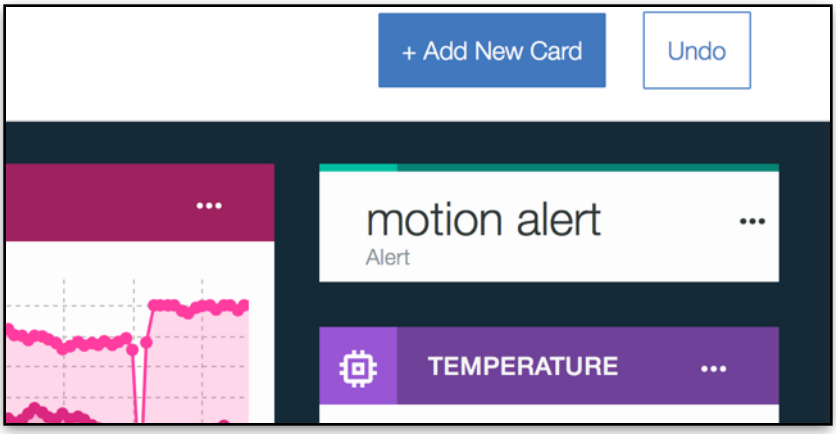
Card 3: Device Alert Text

The last card you will create will not yet have an active data source — the card will display device alerts that will be generated by your analytics application. You can choose to create this card now, or after completing Section 4 (Analyze & Decide).

Following the pattern from the previous 2 steps, create a new card named **Device Alert**, using your device (**simulator**) as the data source. This card will display events from the device named **alert**, with a property **d.text** that will display a string indicating the most recent alert state of the device, computed and published from the analytics application.

Set the card type to a Text Field. Press **Submit**.





In the next section, we will configure, complete and run an analytics application to detect and publish these alert events.

Section 4 - Analyze & Decide

Watson IoT Platform leverages the popular open-standard MQTT protocol for real-time publish/subscribe message between devices and applications. Most streaming analytics tools offer support for the MQTT protocol, making it possible to implement the analytics component of an IoT solution in a variety of ways.

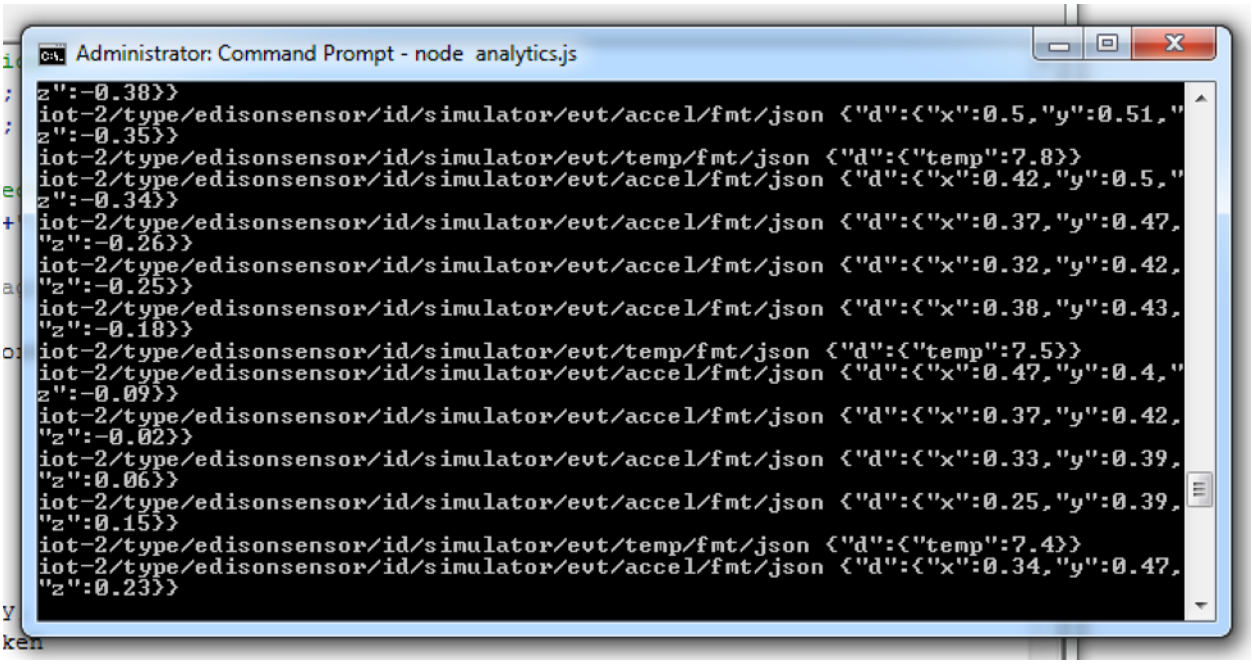
IBM offers a solution (IoT Real-Time Insights, https://console.ng.bluemix.net/catalog/services/iot-real-time-insights/) that integrates easily with Watson IoT Platform to perform contextual and condition monitoring operations on real-time data. However, in this lab you will develop your own lightweight analytics application using Node.js and an MQTT client.

Configure the analytics

The analytics application can be found in **IDA-6158/analytics/**. The application shares the same configuration file you already set up for the simulator, so the remaining steps to run the application are:

1) Open a new command prompt window. Navigate to **IDA-6158/analytics/**
2) Install the required Node.js modules: **npm install**
3) Run the application: **node analytics.js**

If the analytics application has property connected and the simulator is running, you will see some device events printed to the console as MQTT topic and payload.



At this point, take time to open and view the analytics application source code: **IDA-6158/analytics/analytics.js**. The code provided to you connects an MQTT client to Watson IoT Platform using the generated API key and subscribes to all **temp** and **accel** events for your device. These events are stored in memory in the application in a "sliding-window" array.

Two functions, **checkTempThreshold** and **checkAccelThreshold** are empty. Your next step is to implement two simple analytics functions:

1) If the temperature exceeds **tempThreshold** (20) for more than 5 consecutive events, publish an alert event on behalf of this device with the text "temp alert"
2) If the magnitude of the accelerometer reading in the X direction changes by more than 1 in subsequent readings, publish an alert event on behalf of this device with the text "motion alert"

These are realistic scenarios. The first will reduce false positives by ensuring that an event occurs multiple times before an alert is sent. The second is a way to detect when an object has been excessively moved or vibrated.

The functions **checkTempThreshold** and **checkAccelThreshold** are called after receiving a new temp and accel event from Watson IoT. Update these functions in **IDA-6158/analytics/analytics.js** and replace them with the following code:

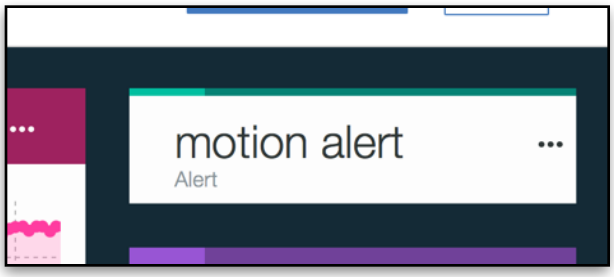
```
function checkTempThreshold() {
	// if the temperature exceeds the threshold for more
	// than 5 consecutive events, send an alert`

	if (tempEvents.length < 5) { return; }

	var overThreshold = true;
	for (var i = tempEvents.length - 5; i < tempEvents.length; i++) {
		if (tempEvents[i].data < tempThreshold) {
			overThreshold = false;
			break;
		}
	}

	if (overThreshold) {
		publishAlertEvent("temp alert");
	}
}
```

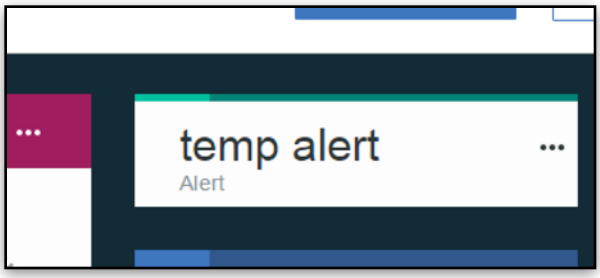
```
function checkAccelThreshold() {
```

```
	// if the magnitude of the accelerometer reading changes
	// by more than 1.0 in subsequent readings, send an alert
	if (accelEvents.length < 2) { return; }

	var thisEvent = accelEvents[accelEvents.length - 1].data;
	var lastEvent = accelEvents[accelEvents.length - 2].data;

	var diff = Math.abs(thisEvent.x - lastEvent.x);
	if (diff > accelThreshold) {
		publishAlertEvent("motion alert");
	}
}
```

Next, restart the analytics application and navigate to the simulator window. “Shake” the Accel - X slider back and forth to generate the “motion alert”, and ensure that you can view this event in the Watson IoT Platform organization dashboard card you previously created.

Then, slide the temperature value above 20 and leave it there for at least 10-15 seconds. Ensure that you can see a “temp alert” in the dashboard.





Next Steps

At this point, you have reached the end of the lab. If you have time remaining, here are a few additional tasks you can attempt:

Extra cards

Follow the steps from Section 3 to create additional cards:

1) A bar graph that shows the “up/down” state of the **input** event
2) A gauge for CPU % that has limits and warning thresholds
3) A line graph that is customized to adjust the sliding window size, show/hide the overview, etc.

Extra alerts

Extend the analytics application to detect and publish a few addtional alert events:

1) A “mag alert”: When the magnometer oscilates by more than X on subsequent events.
2) A “cold motion alert”: A motion alert that occurs when the temperature is less than 5
3) A “konami alert”: An **input** sequence of UP, UP, DOWN, DOWN, LEFT, RIGHT, LEFT, RIGHT, B, A, SELECT

If you require assistance during these steps, please ask an instructor.