**ANDROID-BLUEMIX : TWO WAY COMMUNICATION**

### **Preview**

The main focus of this demo is to develop some good understanding with the IBM Bluemix

Cloud platform and to gain some familiarity with the Android Studio development kit.

The objectives of this demo are:

1] To connect your Android Phone to the IBM Bluemix Cloud.

2] To send a message from our Phone to the Bluemix Cloud.

3] To receive a message from the Bluemix Cloud to our Phone.

4] To sync our Twitter account with the Bluemix and to receive a message in our Phone whenever a tweet occurs for the mentioned topic.

**Prerequisites**

1. **Install Java in your system**

Check your Java version using:

java -version

We require java version to be 1.8 or above.

Install Java 1.8 from : <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Note down the location where java is installed, i.e. the path to “jdk1.8.0\_71”. We will have to provide its path in Android Studio.

If you are on Linux and already have Java 1.7, you can update to 1.8 using the instructions given on <http://stackoverflow.com/questions/30177455/moving-from-jdk-1-7-to-jdk-1-8-on-ubuntu>

1. **Install Android Studio in your system**

For Windows/Mac, download Android Studio from <https://developer.android.com/studio/index.html>

For linux, install ubuntu make and android from

<https://wiki.ubuntu.com/ubuntu-make>

in the “How to use it” section near the bottom. For an Ubuntu version less than 16.04, you may have to use PPA. If “umake android” does not work, you can follow the commands given at <https://itsfoss.com/install-android-studio-ubuntu-linux/>. It gives an alternative way to install using PPA.

1. **Enable USB Debugging in your Android Phone**

Follow the instructions given at <https://www.kingoapp.com/root-tutorials/how-to-enable-usb-debugging-mode-on-android.htm>l

Connect your Android to your laptop.

1. **Basic Hello World Application using Android Studio (extra...not required to be performed)**

Create new project

Basic Activity template

app>src>main>java>com….>MainActivity.java

Run it

Wait for few seconds...In your Mobile HelloWorld! Will display

### **Building a sample app that communicates with Bluemix**

1. **Downloading the code**

**Steps:**

* Download the code from <https://github.com/ibm-watson-iot/iot-starter-for-android?cm_mc_uid=08500001042914672352114&cm_mc_sid_50200000=1467746395> as a zip or use “git clone https://github.com/ibm-watson-iot/iot-starter-for-android.git” if you are comfortable with git. Extract the zip.
* Open the project from Android Studio. Go to File -> New -> Import Project. Choose the folder that we downloaded above.

**2. Fixing Build Errors**

**Steps:**

* Android Studio automatically starts building the project when you import it. The build will fail as the project was originally created user older build tools and SDK.
* Click on File -> Project Structure. Under “Modules” on the left, choose the project that we opened ( iot-starter-for-android-master).
* Click on “Build Tools Version” drop down and choose the version installed (e.g. 24.0).
* Click on “Compile SDK Version” and choose the same version that you chose above ( e.g. 24.0). Click on “Ok”.
* Click on “Projects” tab on the left. Expand “Gradle Scripts” and open “build.gradle” file. Remove the lines “android { compileSdkVersion 24 buildToolsVersion **'24.0.0'** }”
* Expand “iot-started-for-android-master” on the left, expand “app” folder and choose “build.gradle”.
* Under “android”, change the “compileSDKVersion” to the version you selected in the “Project Structure” (e.g. 24).
* Under “android”, change the “buildToolsVersion” to the version you selected in the “Project Structure” (e.g. 24).
* Under “defaultConfig” change the “targetSDKVersion” to the above SDK version. ( e.g. 24)
* Under “dependencies” change the version in “compile **'com.android.support:appcompat-v7:21.0.3’** E.g. change it to compile **'com.android.support:appcompat-v7:24.0.0’**
* Click on “Try again” at top left to run Gradle Sync again.
* There may be a sync error, in which case, click on “Fix Gradle Wrapper and re-import project”. Click on “Update” when prompted.
* Gradle Sync should finish without errors.

**3. Registering the Android Device in Bluemix.**

**Steps:**

* Login to your IBM Bluemix account.
* Open any existing “Internet of Things Foundations” service. If you do not have one, create by going to “Catalog” -> “Internet of Things Platform Starter”.
* Open the Internet of Things Service and click on “Launch Dashboard”.
* Click on devices icon on the left and then click on “Add a device”.
* Create a new device type “Android”. **The spelling and case must exactly match** as the Android app has this device type hard coded in its code.
* Add a device of the device type “Android”.
* Note down “Organization” and “Auth Token”.

**4. Running the Android App to communicate with Bluemix.**

**Steps:**

* In Android Studio, run the app. Keep the phone connected to your laptop.
* Choose your device in the “Select Deployment Target” window and click on “Ok”.
* The application will build and take 3-4 minutes. Then the APK will be installed on your mobile.
* If you are unable to run the app, you can also install the APK directly by the following

steps:

1. On your phone, go to Settings > Security. Under Device Administration,enable “Unknown sources”. Now you can install .apk files from outside of Google Play.
2. Open the browser on your phone, and enter this URL: <https://ibm.box.com/iotstarterapp> . Download the APK. Install it by opening it.

* When the app runs, you will see a screen like -

|  |  |
| --- | --- |
|  |  |

* Enter the organization name and authentication token. And click on “Activate Sensor”.
* Your app will start sending acceleration data. If it does not, check the authentication

Token.

* Click on the menu icon on top left. Click on “Open Profiles” and “Save Settings”. Now you

will not have to enter the details when you close and launch the app later.

**Verification:**

1. Open IBM Bluemix Internet of Things dashboard. Open your device and see the

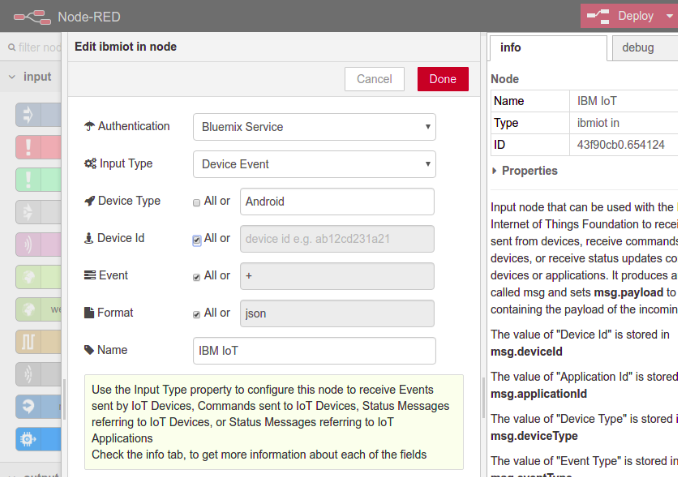
messages.

**5. Communicating from Bluemix to the Android App**

**5.1 Logging all messages in Node Red**

**Steps:**

1. Keep the Android App running.
2. Open your “Internet of Things” service and go to “App Overview”.
3. Open the app url and then open the Node Red editor.
4. Create a new flow.
5. Drag and drop an “IBM IoT In” block.
6. Open the node and change “Authentication” to “Bluemix” service.
7. Keep “Device Input” as “Event”.



1. Enter “Device Type” as “Android”. Deselect the “All” checkbox.
2. Select the “All” checkbox for “Event”, “Format” and “Device Id”.
3. Connect the output to a “debug” block.
4. Deploy the Node Red app.

**Verification:**

1. Accelerometer messages should be visible in debug tab.

**5.2 Logging text messages in Node Red**

**Steps:**

1. In the “IoT IBM In” node, deselect “All” checkbox for “Event” and enter “text” in the textbox.
2. Deploy the app.

**Verification:**

1. In the Android App, click on “Send Text” button and enter any text.
2. Verify that you can see the message in “debug” tab.

**5.3 Sending messages back to the Android App**

**Steps:**

1. Connect a function node to the output of the “IBM IoT In” block.
2. Enter the below code :

// Get the text sent by Android App and append “ from Bluemix” to it.

var text = msg.payload.d.text + " from Bluemix.";

// Command type. Android App is listening to “text” command.

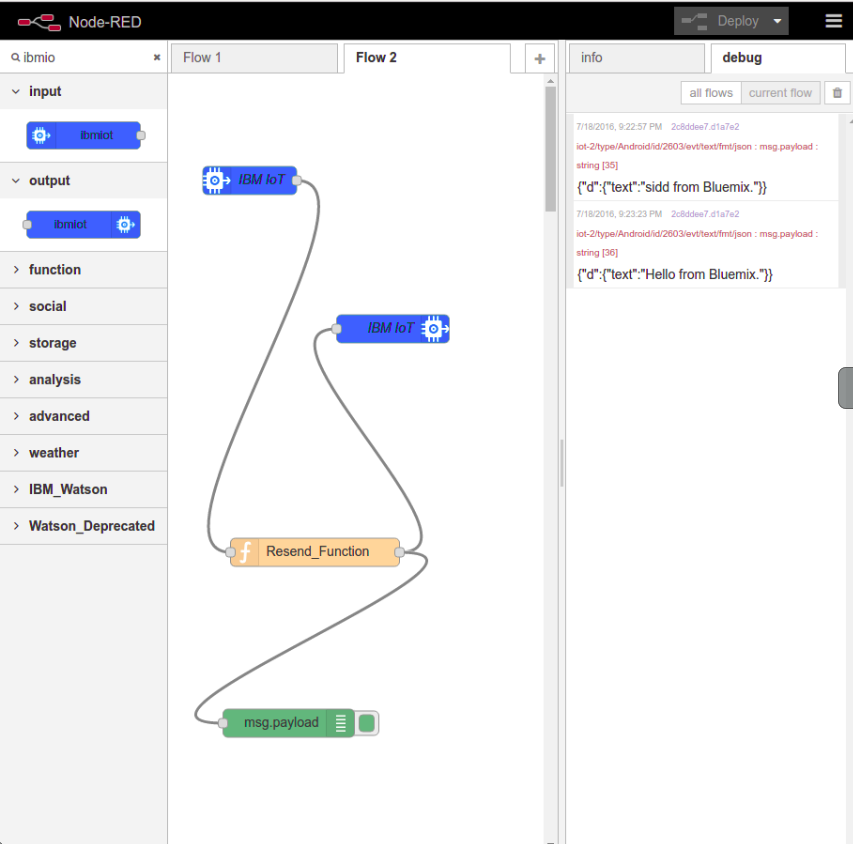
msg.eventOrCommandType = "text";

// Add the text to the payload.

msg.payload = JSON.stringify({"d":{"text":text}});

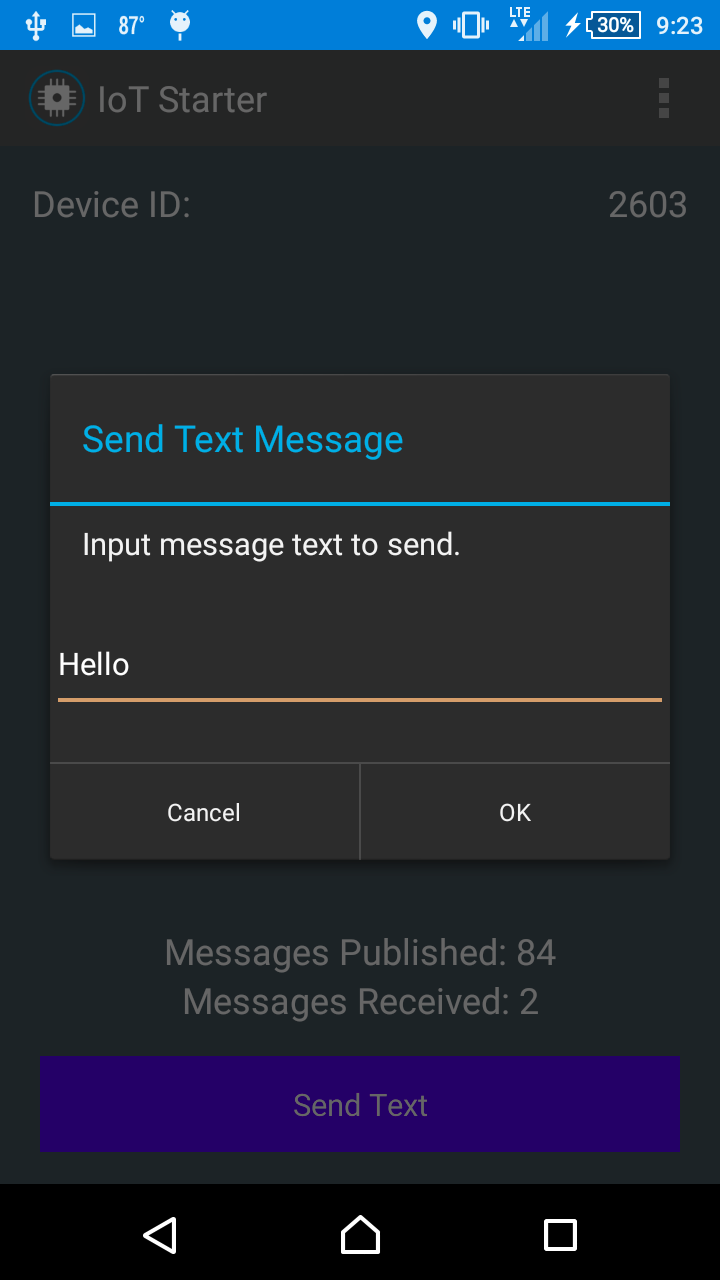
return msg;

1. Drag and Drop an “IBM IoT Out” node .Connect the output of the function block to the input of the “IBM IoT Out” node.
2. Double click on the “IBM IoT Out” block. Choose “Bluemix Service” as the “Authentication”.
3. Choose “Device Command” as the “Output Type”.
4. Enter “Android” as the Device Type.
5. Enter your “Device Id”.
6. Enter “text” as the “command type”.
7. Enter “json” as the “format”.
8. Enter “{"d":{"value":"text"}}” as “data”.
9. Now stop the Android App.
10. Open the “Message Conductor” (app-->java-->utils-->Message Conductor) file and go to “steerMessage()” function. This function is called whenever the App receives a message on a topic it has subscribed to.
11. Go to the case for “Text\_Event”.
12. Enter the line Toast.makeText( context, messageText, Toast.LENGTH\_LONG).show(); Enter this below app.getMessage(). … line.
13. This will display a notification of the received message for 5 seconds on the Android App.
14. Now deploy the App.



**Verification:**

1. Send a text message from Android.



1. You should see the same message back in your app, that is sent by Bluemix.
2. Swipe the screen of your App to your left to see the message log.



### **Sending messages from Twitter to the Android App**

**Steps:**

1. In the Node Red editor, open the same flow that you created above.
2. Drag and drop a Twitter node.
3. Double click on the “Twitter” node and authenticate.
4. Enter “#EB2” in for.
5. Add a “function” block.
6. Enter the code as

// Get the tweet message

var tweet = msg.payload;

// Get the user

var user = msg.tweet.user.name;

msg.eventOrCommandType = "text";

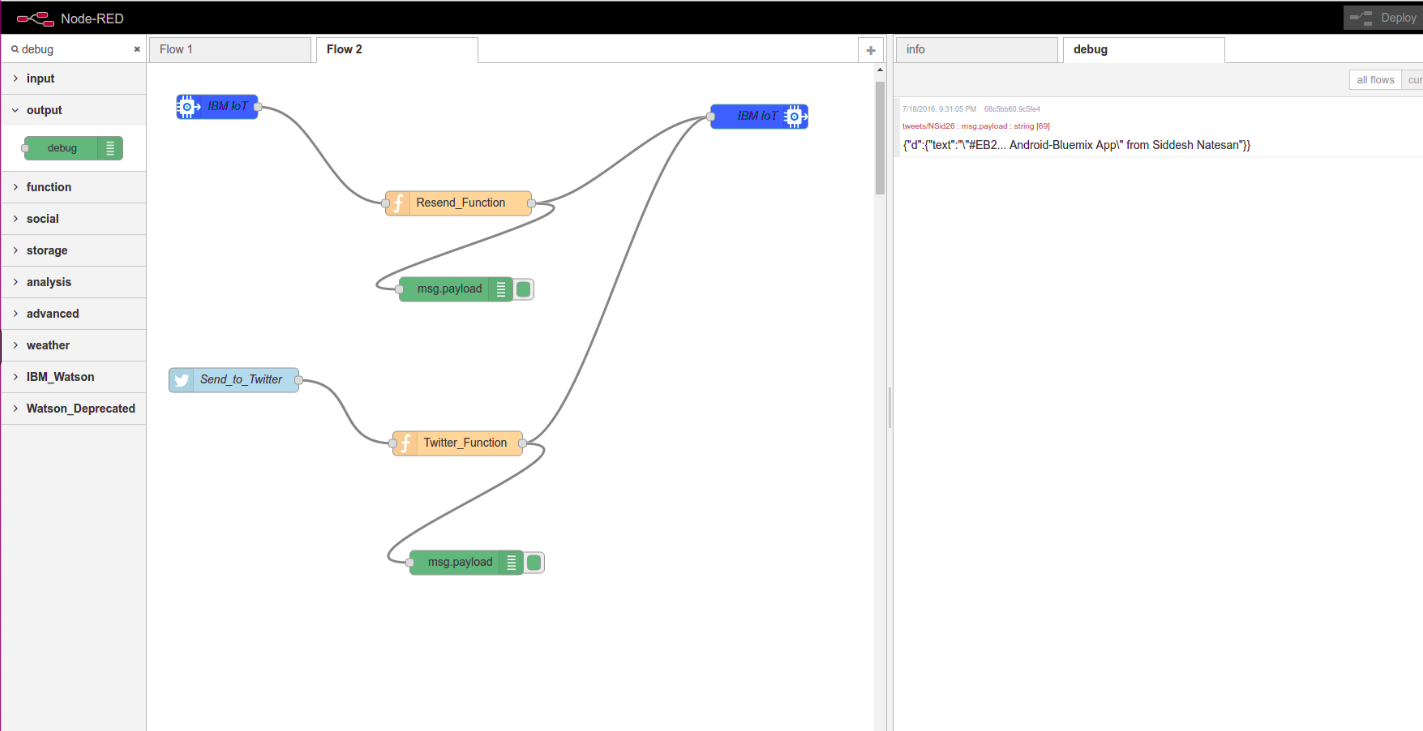
// Create the notification text

var notification = '"' +tweet + '" from ' + user;

msg.payload = JSON.stringify({"d":{"text":notification}});

return msg;

1. Connect the output of Twitter block to input of function block and output of function block to input of IBM IoT Out block
2. Deploy the app.

****

**Verification:**

1. Tweet a message with “#EB2”.
2. You should get a notification.

