IOT GUIDE FOR **APP DEVELOPMENT** AND **CLOUD COMMUNICATION**

University of Florida MIST Makers

Fall 2016

Written by Dean Fortier

In this guide I will show you what we did in the IoT (Internet of Things) aspect of our project. We used IBM's Bluemix Platform (for IoT services) and EvoThings Studio (for app development). There are many *better* ways to achieve the same goals which you should explore.

Acknowledgements/Message

This tutorial came from the SmartGarden prototype project and for the full project overview, look at Cody Rigby's guide for help with the hardware design and the TI CC3200

Much Thanks to Cody Rigby, Chris Falck, and Daniel Tola for contributing to this part of the project.

If you have any general questions, feel free to text or call me at (863) 221-4094. If you have any coding questions Chris Falck may be able to help. He is a wizard. Find us on Facebook.

I am not the best at coding or understanding what is going on in the background, but I learned enough to know how to use what I have below. This guide is meant to be a stepping ground so you are not lost in the same places we got lost.

All of the code I used in HTML for our prototype is at https://drive.google.com/drive/u/0/folders/0BzLmnQtnLIaFTGtMRlk2UGxoSG8 I recommend making your own because of how messy and unorganized mine is.

Table of Contents

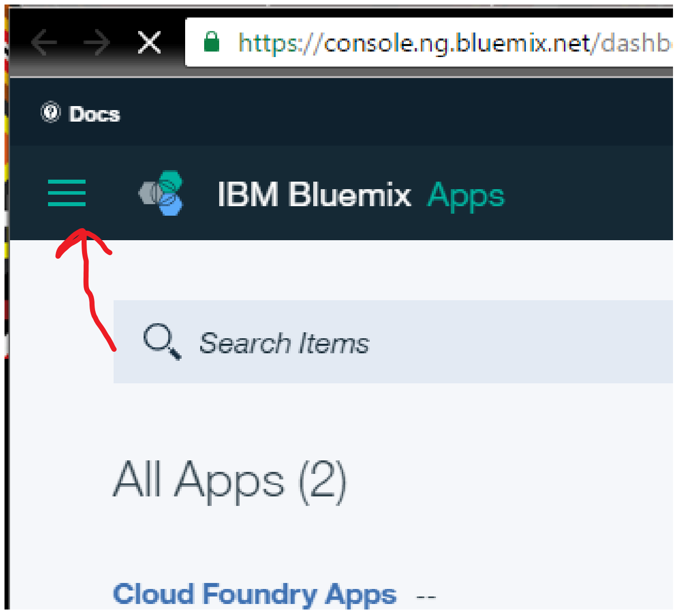
I. Basics

a. What is IBM Bluemix?

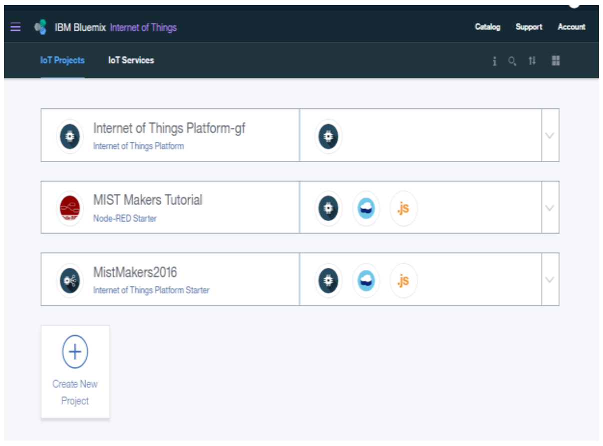
Bluemix is a platform used for IoT. They have many **services** and **apps** available to make an IoT project a success.

I recommend you make a practice account to try some of the things described below. The free account lasts 30 days, so learn what you need within that time frame before using the final account. There are so many things to explore and use on the Bluemix site, but we are just going to use a few parts of Bluemix.

Once logged in with your new account name, click on the **hamburger menu** at the top left of the screen



Then click on **services**, then **Internet of things.** This should navigate you here.


IBM Bluemix Internet of Things
Catalog
Support
Account
IoT Projects
IoT Services
Internet of Things Platform-gf
Internet of Things Platform
MIST Makers Tutorial
Node-RED Starter
MistMakers2016
Internet of Things Platform Starter
Create New Project

(looks something like this)

This is where you will make a project and add services and apps to that specific project.

a.i. Devices

“Devices” is a Bluemix service where you can set up devices that you can later manage in the Bluemix platform. I will go over this later.

a.ii. Cloudant

“Cloudant” is a Bluemix service that you can send information to, or a Cloud. Once you have a Bluemix account, you automatically have a Cloudant account. Inside you can create multiple databases. Inside those databases is where you would send the data in packets that are saved as “documents” that each have a specific document ID.

For Example, let's say that I want to send the current temperature and humidity to a database called "weather" in my Cloudant account. I would decide to send up the temperature and pressure every 10 seconds. Every 10 seconds I would create a new document in the "weather" database with its own document ID. It would send up something that looks like:

```
{
  "_id": "007e4a40b4eb73aaef4433d3865a9aea",
  "_rev": "1-0fc1c7a50f7715609cb948e52d06cf6f",
  "topic": "iot-2/type/CC2650/id/weatherDevice/evt/status/fmt/json",
  "payload": {
    "Temperature": "71.83",
    "humidity": "40.66",
  },
  "deviceId": "weatherDevice",
  "deviceType": "CC2650",
  "eventType": "status",
  "format": "json"
}
```

I will describe later how to get this information and how to later use it.

Fun Fact: Cloudant is a NoSQL database system because it uses document IDs versus an SQL server which uses rows and columns to categorize items.

Maybe look into using a different server or an SQL server for a different project.

a.ii. Node-Red

To be honest I don't know many of the capabilities of Node-Red. All I used it for was taking Data from a **Device** and moving it to a **Cloudant** database. Node-Red is a GUI interface for Node.js but that is about all I know. Look up how to use Node.js in JavaScript if you want to be a real pro in IoT.

b. What is EvoThings?

EvoThings is a program that lets you view HTML programs on your phone. EvoThings can do this very fast so you can see what the app will look like on your phone as you program.

REASONS I LIKE EVOTHINGS:	**REASONS I DISLIKE EVOTHINGS:**
• Rapid Prototyping (much faster than developing in Android Studio) • HTML/ JavaScript is easy to use and easy to learn with help of Google. • EvoThings has awesome examples and templates that you can get started on. • Decent community support	• Need to deploy apps from computer to phone (phone is not standalone) • Not a real, downloadable app • Not easy to transition EvoThings app to market app (but it is possible)

Maybe for your project you will decide that you want to develop on Android Studio (uses Java) for a better final product.

b.i. HTML

HTML is the standard programming language for web pages. If you have Chrome, open up a webpage and hit Ctrl+Shift+I and you may be able to find the html file and you will also be able to debug as well. There is a ton of support and tutorials online for HTML. HTML is also the language used by EvoThings.

b.ii. JavaScript

JavaScript is a powerful language used in HTML. You can change and define variables and functions in JavaScript, then display with HTML. Look up “how to use JavaScript in HTML” for help.

b.iii. HTTP

HTTP is an application protocol. We can use HTTP in JavaScript (as you will see later). We can do GET, HEAD, POST, DELETE requests using HTTP.

The way we used HTTP in our project was to GET data from document IDs in our cloud databases.

b.iv. MQTT

Not as popular as HTTP, but still another application protocol that only has publish/subscribe as commands. It is very lightweight. I don’t understand all of what goes into it.

There are MQTT Brokers (or servers) which allow communication between clients (the Devices service on the Bluemix Platform).

We used MQTT to publish sensor data to Bluemix’s Device service.

c. Pros and Cons of Bluetooth and WiFi

These may seem obvious, but these are all things I found during this project. I used a template from EvoThings for my Bluetooth communication, and I relied on my phone having internet connectivity for the rest of the project. Cody Rigby is more familiar with WiFi connectivity.

Bluetooth is fast, free, and easy to prototype with once connected. Limited to communication with phone.

WiFi needs a dedicated router when using on UF campus, not always free, but once connected, you can connect directly to an online server.

II. How to send Data to Bluemix

This is the tutorial that I used to implement MQTT

https://evothings.com/publishing-sensor-data-to-ibm-watson-iot-platform/

if the link is not up, text me. I have the link downloaded as a PDF. (this is what I first used)

https://github.com/hammadtq/Evothings-Demo-Apps/tree/master/bluemix-iot-cloud-CC2650

above is the github code for the example that I used and modified.

Make sure you also have Paho Javascript client library downloaded to your EvoThings folder

Use this link to download the Paho Library: https://github.com/eclipse/paho.mqtt.javascript

I believe you actually do not have to download, but you can use this script tag in your program that implements the MQTT functions. (index.html for us)

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/paho-
mqtt/1.0.1/mqttws31.js" type="text/javascript"></script>
```

Follow this link if you want to send data from the TI CC2650 SensorTag to Bluemix via MQTT protocol.

To go more in depth with MQTT and Bluemix, Bluemix has documentation on how to publish and subscribe. I recommend you look at app.js in the project folder that I provide of the code to see an in depth example of publishing.

Bluemix Documentation will contain all documentation for connecting devices.

Bluemix MQTT Documentation can be found from previous link.

III. How we pushed data from Bluemix to the Cloud

Once you have data published data to a device that you made in the last step, open Node Red in your platform.

(I apologize I cannot do screenshots, my Bluemix account free trial expired)

You may need to create a Cloudant service (similar to the Device service)

Have the input be the device you pick, and the output be the Cloudant database (there are 2. Try both)

This will create a new database with a name you choose.

Each upload will consist of a new document ID

You are done! You now have a cloud server that has stored information.

To access your different Cloudant databases, go to your project Platform on Bluemix and click on the Cloudant Service.

If you do not feel satisfied with uploading information this way through MQTT, and IBM Bluemix, log in to Cloudant and look in the documentation to see about PUTting data via HTTP protocols instead. I did not put data on my servers this way, but the HTTP ways may allow you to bypass Bluemix completely for quicker upload times (maybe). To do this you will need to learn HTTP requests. I will show you the GET request later.

Cloudant Documentation

Cloudant HTTP Documentation

IV. How to Retrieve Data from the Cloud

We used a GET request with the specific URL from the Cloudant website.

Here is a URL we used a GET request to get information used in the SmartGarden Project:

https://3e0b8d85-cd9f-4947-8543-f30b652dd4c8-bluemix.cloudant.com/deansjunk/007e4a40b4eb73aaef4433d3865a9aea

or

https://3e0b8d85-cd9f-4947-8543-f30b652dd4c8-bluemix.cloudant.com/bigimage/_all_docs

(try pasting in an "id" to replace the "_all_docs" in the URL and see what you get)

Ex:

https://3e0b8d85-cd9f-4947-8543-f30b652dd4c8-bluemix.cloudant.com/bigimage/2d1e531cd167cd6ca8ac3dbdf0b5eaff

The tutorial link below gave me a basic understanding of HTTP requests. More specifically the GET request.

https://www.kirupa.com/html5/making_http_requests_js.htm

I recommend googling something like "making http requests in [insert programming language here]" to learn the ins and outs of HTTP requests.