Workshop: Connecting IoT devices to the cloud with IBM Watson IoT and mbed Cloud

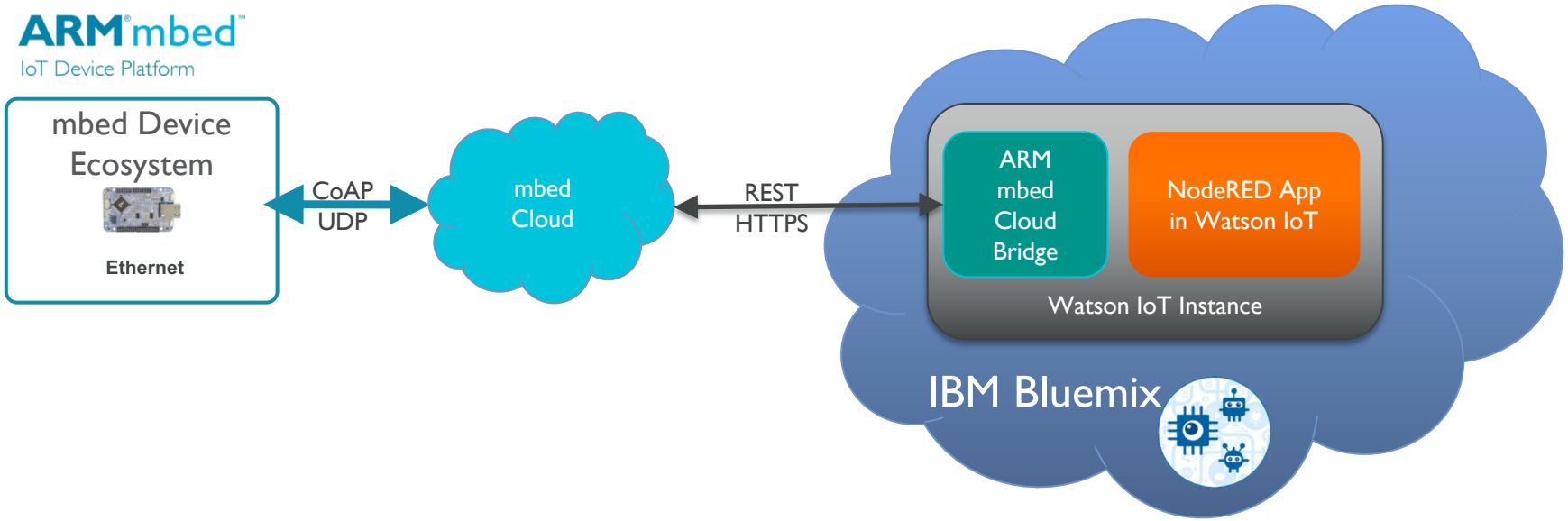
ARM

Doug Anson
Solutions Architect / IoT BU / ARM

Brian Daniels
Applications Engineer / IoT BU / ARM

April 26, 2018 – v2.5

What will we build in this Workshop?



- Connect your mbed device into Watson IoT through mbed Cloud and Watson's Cloud Bridge
  - Create a simple mbed device and connect it to mbed Cloud
  - Watson IoT now has a fully integrated mbed Device Cloud Bridge that links mbed Cloud to Watson!!
  - Exploration of the device data telemetry using NodeRED flows within Watson IoT

Workshop: Let's get started!

- Create and setup your Bluemix account (should be completed prior to workshop)
- Install the necessary tools/drivers into your Windows/Mac/Linux PC (should be completed prior to workshop)
- Create mbed developer and mbed Cloud accounts (should be completed prior to workshop)
- Retrieve a set of provisioning credentials (mbed_cloud_dev_credentials.c)
- Import our mbed sample project into the online IDE
- Update the sample project with the provisioning credentials (mbed_cloud_dev_credentials.c)
- Compile, Install, Download, and Copy into the mbed device
- Connect a Serial Terminal (115200,8N1, proper mbed COM port chosen for Windows users…)
- Send the "Break" command to reset the mbed device
- See the device output on our Serial Terminal (PTSOOI method...)

NOTE: During the workshop, be sure to double-check your copy/paste operations...

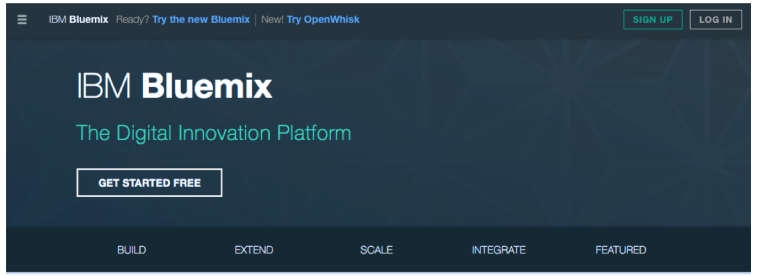
Quick Links: Visit https://github.com/ARMmbed/bridge_workshop_ibm_watson_cloud

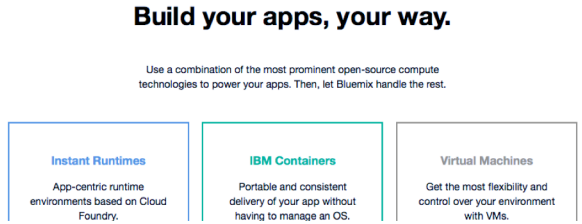
- README.md has most of the links in the workshop...

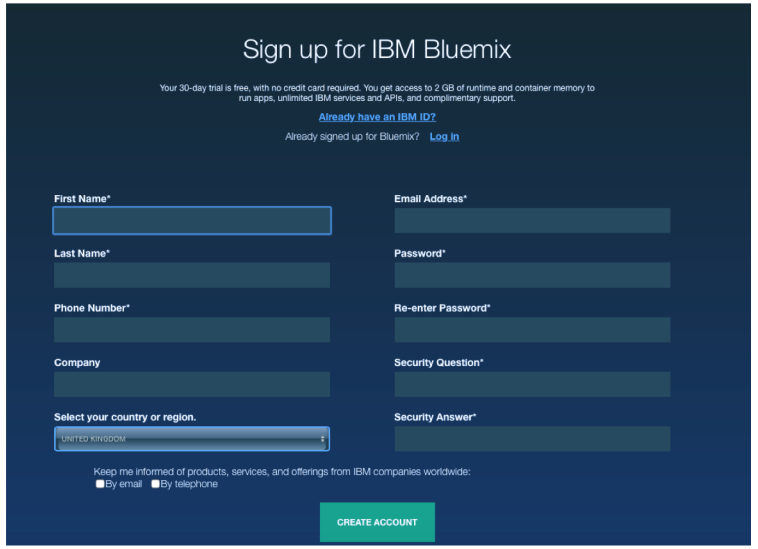
Create our Bluemix Account

- Navigate to the Bluemix Dashboard: https://console.bluemix.net/
- Press "Sign Up"
- Complete the sign-up process
- A confirmation email will be sent that must be acknowledged
- Log into your Bluemix account and establish your default organization and space

Prior to Workshop



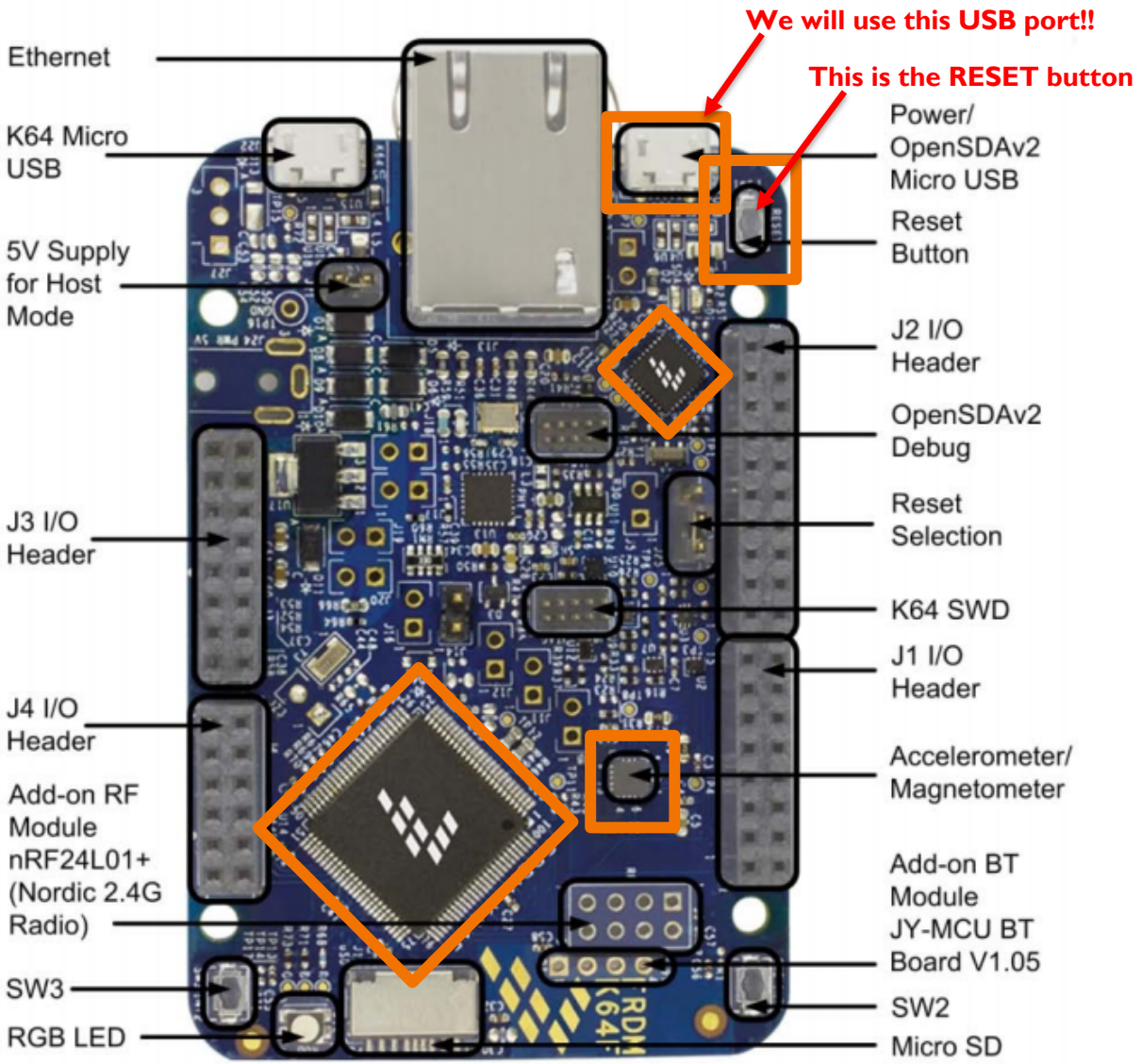




ARM

NXP- FRDM-K64F Overview

- **Freedom Development Platform - Quick, simple development experience with rich features**
  - Cortex-M4, 120MHz, 1MB Flash, 256KB SRAM
  - Easy access to MCU I/O
  - 3-axis accelerometer/3-axis magnetometer
  - RGB LED
  - Add-on Bluetooth Module
  - Built-in Ethernet/Add-on Wireless Module
  - Micro SD
- **Arduino** shield compatible
- Flash programming functionality
- Enabled by OpenSDA debug interface

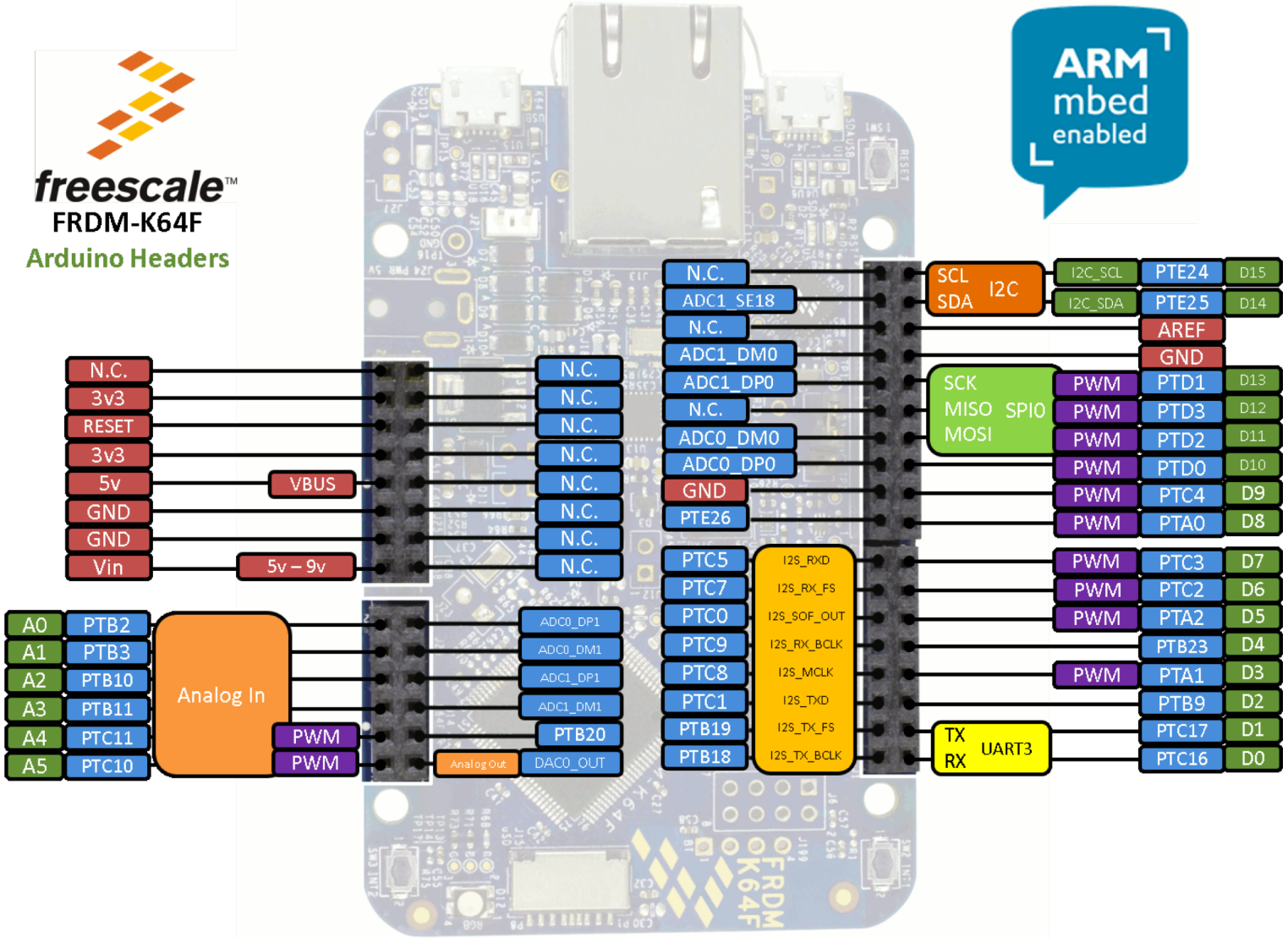


ARM

freescale™

FRDM-K64F

Arduino Headers



ARM

Install the Necessary Tools

Prior to Workshop

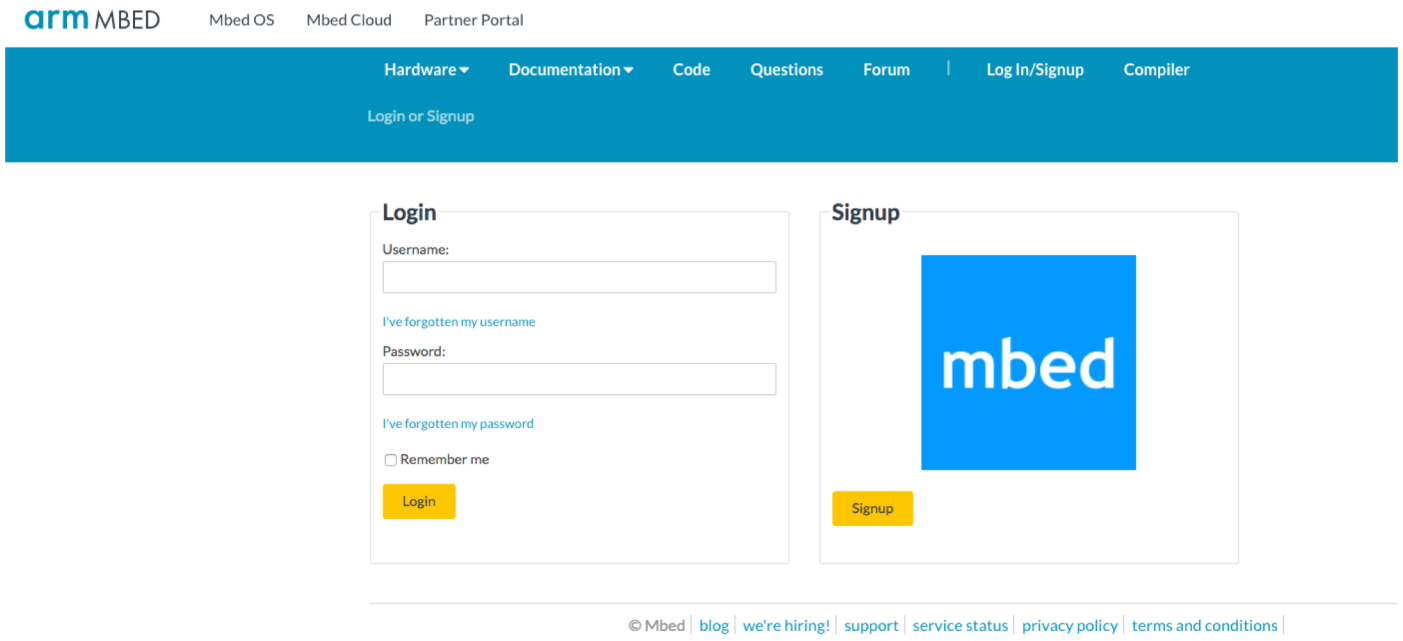
- **Windows**
  - **IMPORTANT**: Install the mbed USB Serial driver - https://os.mbed.com/media/downloads/drivers/mbedWinSerial_16466.exe
    - Insert the USB cable and mbed device BEFORE running the Serial Driver installation…
    - the installer MUST see the device *first*
  - Serial Terminal: Putty - http://www.putty.org/
  - Option: Install CoolTerm - http://freeware.the-meiers.org/CoolTerm_Win.zip
  - Chrome and Firefox Browsers installed – NOTE: IE will NOT WORK… Please use Chrome and/or Firefox
- **Mac**
  - Serial Terminal: CoolTerm - http://freeware.the-meiers.org/CoolTerm_Mac.zip
  - Chrome and Firefox Browsers installed
- **Linux**
  - Serial Terminal: CoolTerm - http://freeware.the-meiers.org/CoolTerm_Linux.zip
  - Chrome and Firefox Browsers installed

Connect both your USB cable and Ethernet cable to the K64F and leave it there for now

Create Your mbed Account

Prior to Workshop

- Navigate to: https://os.mbed.com/account/login/?next=/
- Create an Account

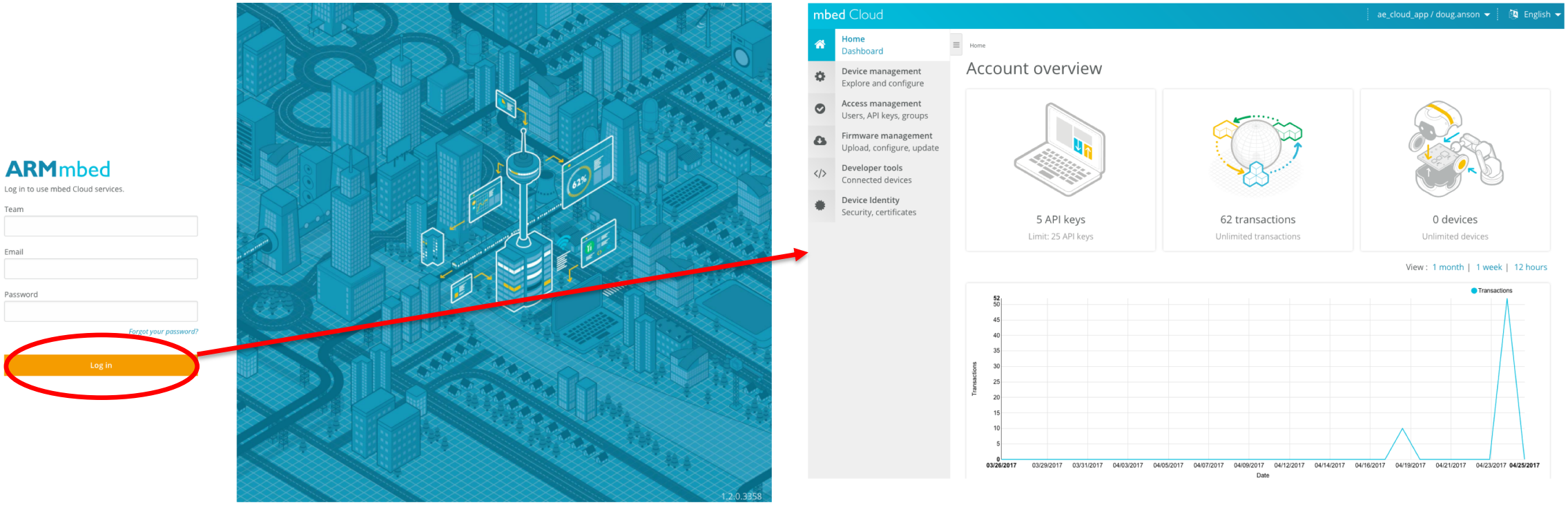


ARM

Create Your mbed Cloud Account… Prior to Workshop

- Navigate to the mbed Cloud Dashboard: https://portal.us-east-1.mbedcloud.com

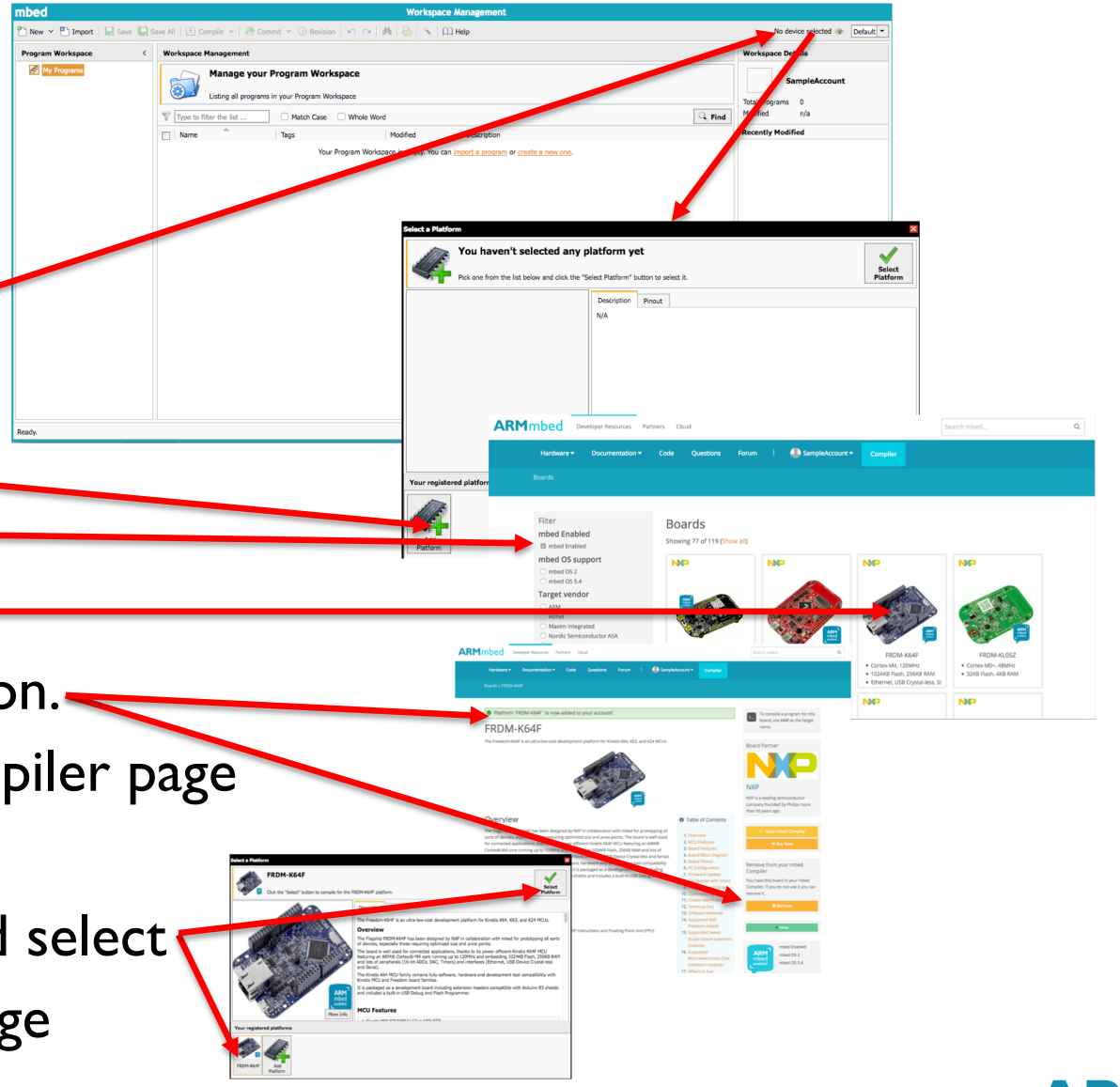
- Confirm that you can log into your mbed device mbed Cloud dashboard



Log into the Online IDE – Add the K64F Compile Target

- Go to https://os.mbed.com/

- Select the "Compiler" page
- Click on "No device selected"
- Click on "Add Device"
- Check "mbed Enabled"
- Click on the "FRDM-K64F"
- Click on "Add to Compiler"... note addition.
- Dismiss all windows... go back to the compiler page
- Click on "No device selected"
- Now, select the "FRDM-K64F" to add and select
- Dismiss and confirm on your compiler page

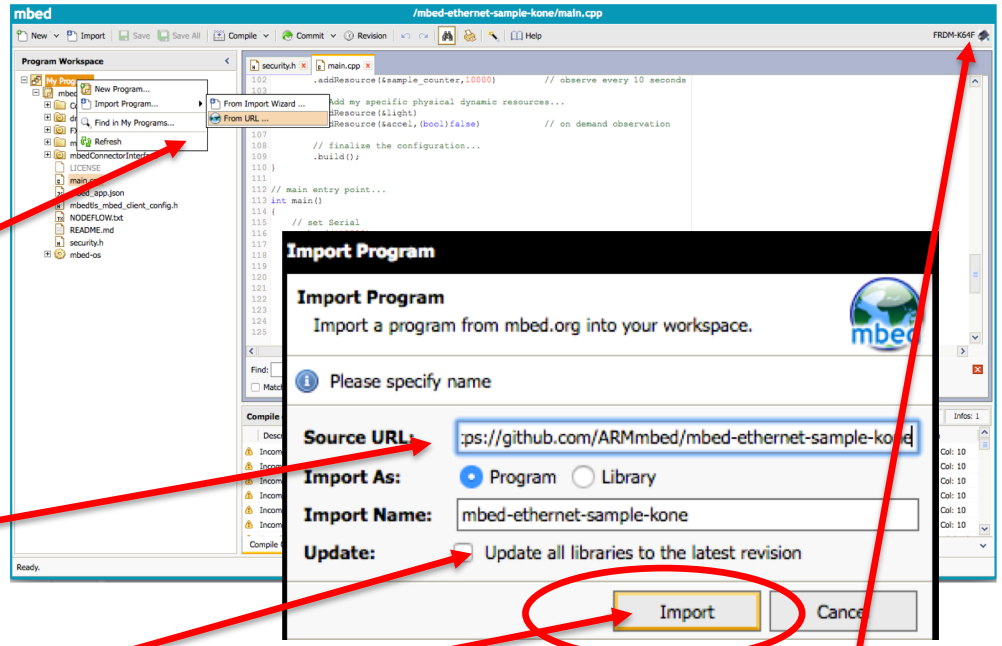


ARM

Import our K64F Endpoint Project

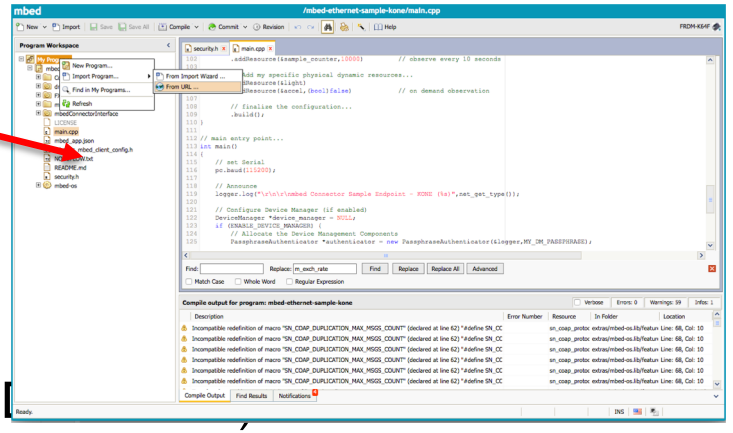
- Go to https://os.mbed.com/
- Select the "Compiler" page
- Right-click on "My Programs"→"Import Program"
- Choose Select "from URL…"
- Enter this URL (Leave the "Update all libraries…" **unchecked**)

  https://github.com/ARMmbed/mbed-cloud-sample/
- Press "Import", the ensure that "FRDM-K64F" is selected



Set Provisioning Credentials in Your Endpoint Code

- Go back to the mbed Device mbed Cloud dashboard: https://portal.us-east-1.mbedcloud.com
  - In the left sidebar, select "Device Identity"→ "Certificates"
  - Under "Actions", select "create a developer certificate"
  - Give your developer certificate a name and description… press "create certificate"
  - Press "Download Developer C file" – this will deposit "mbed_cloud_dev_credentials.c" in your downloads directory
- Now, go back to the Compiler page of your online IDE
- Replace mbed_cloud_dev_credentials.c with newly downloaded one from the dashboard
- Save
  - Glance at main.cpp… a clean and simple mbed endpoint example
  - Exposes two CoAP resources: accelerometer and LED
- Select the project name and press the "Compile" button
- The endpoint code should compile up successfully
- The online IDE will deposit a "bin" file into your downloads directory
- Drag-n-Drop this bin file to your "MBED" flash drive (may also be called "
- K64F green LED will flicker for a bit, then stop, and dismount/remount…

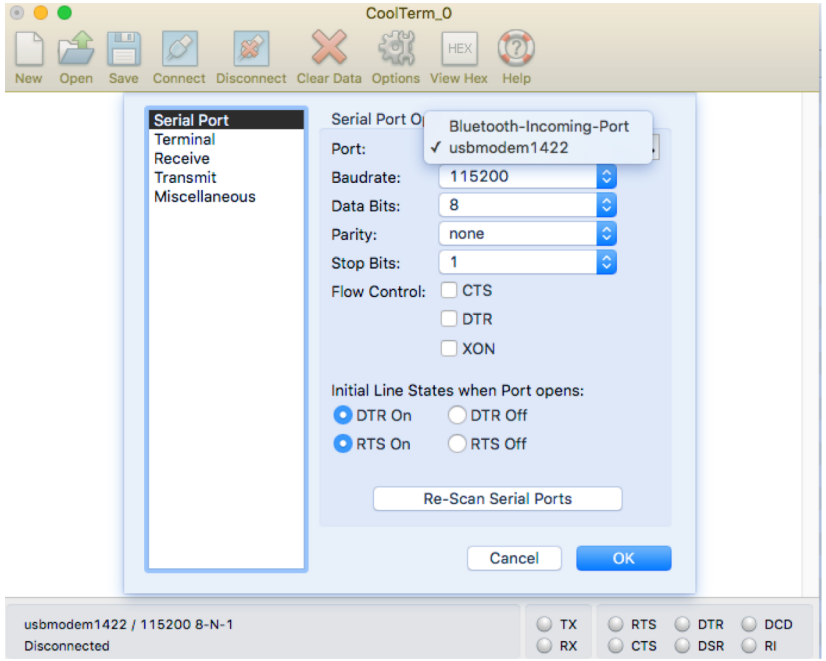


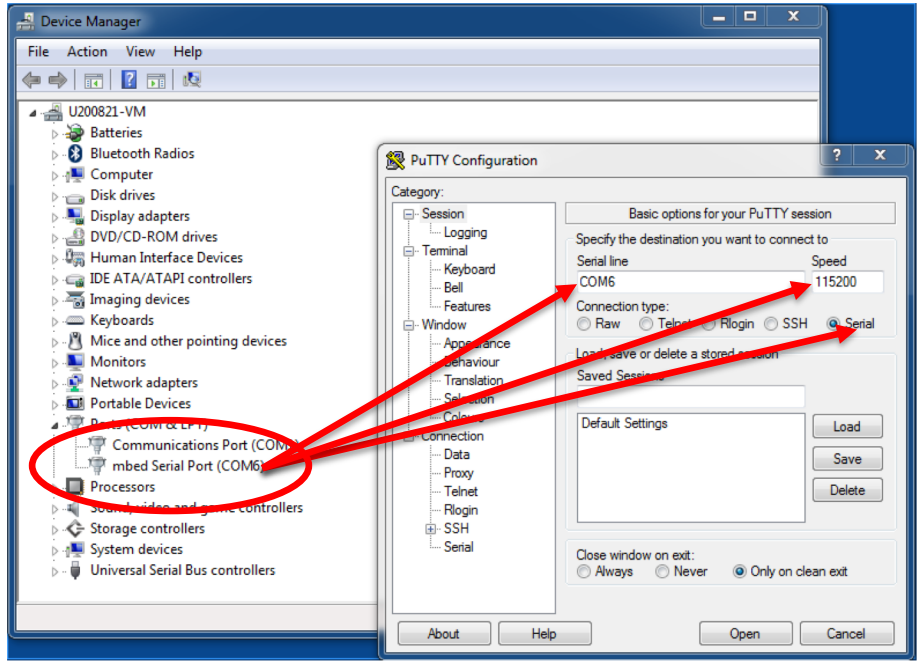
ARM

Running your Endpoint Code

- Bring up your Serial Terminal
  - CoolTerm for Windows, Mac, Linux: Select: "Options→"Re-scan serial ports". Select the mbed one found.
    - For MAC, you may need to "authorize" the CoolTerm Application under your "System Preferences"-> "Security & Privacy"… you may have to allow apps to run from "any developer", then authorize the launch of CoolTerm.
  - PuTTY for Windows : You must determine what COM port your mbed device is. Look in the Windows Device Manager FYI

- For the endpoint serial configuration, set the baud rate to 115200 baud, defaults for everything else: (8, N, one)
  - PuTTY for Windows: Ensure that you have "Serial" radio button selected…



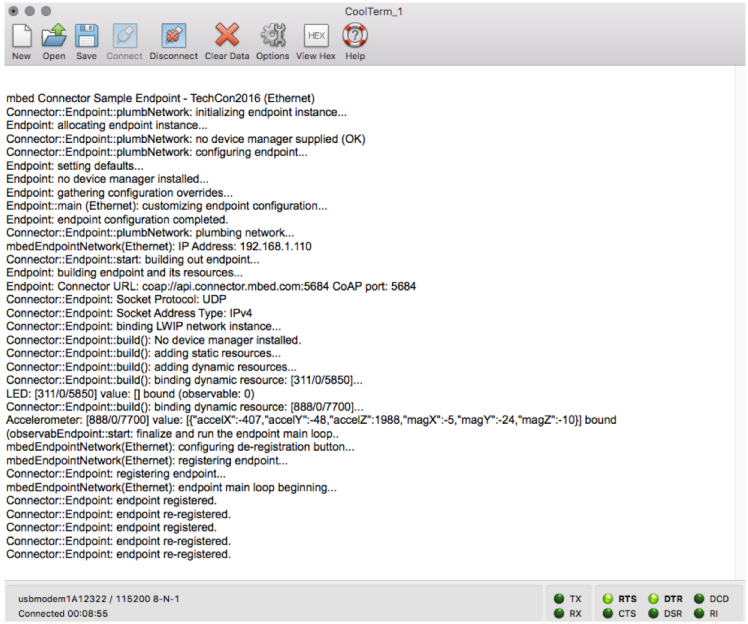


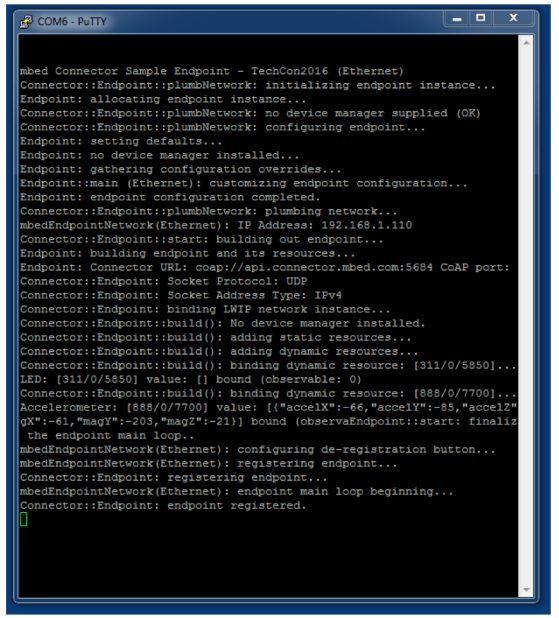
Running your Endpoint Code…

- Connect your Serial Terminal to the K64F:
  - CoolTerm for Windows, Mac, Linux: Simply press the "Connect" button on the top part of the CoolTerm GUI
  - PuTTY for Windows: Press the "Open" button

- Send a "Break" command from the serial terminal (or press the RESET button on the K64F)
  - CoolTerm for Windows, Mac, Linux: "Connection" → "Send Break"
  - PuTTY for Windows: Right-click on top of Window, Select "Special Command" → "Break"

- Look at the output – you need to confirm that you see "endpoint registered" in the output:





Status Check

So far we've completed the following

- Setup our accounts and PC with appropriate tools (prior to workshop)...
- Retrieved a set of provisioning credentials (mbed_cloud_dev_credentials.c)
- Imported our mbed sample project into the online IDE
- Updated the sample project with the provisioning credentials (mbed_cloud_dev_credentials.c)
- Compiled, Installed, Downloaded, and Copied into the mbed device
- Connected a Serial Terminal (115200,8N1, proper mbed COM port chosen for Windows users…)
- Sent the "Break" command to reset the mbed device
- Saw the device output on our Serial Terminal (PTSOOI method...)

Next, we will import and configure the Watson IoT mbed Cloud Bridge...

Watson IoT mbed Cloud Bridge Configuration

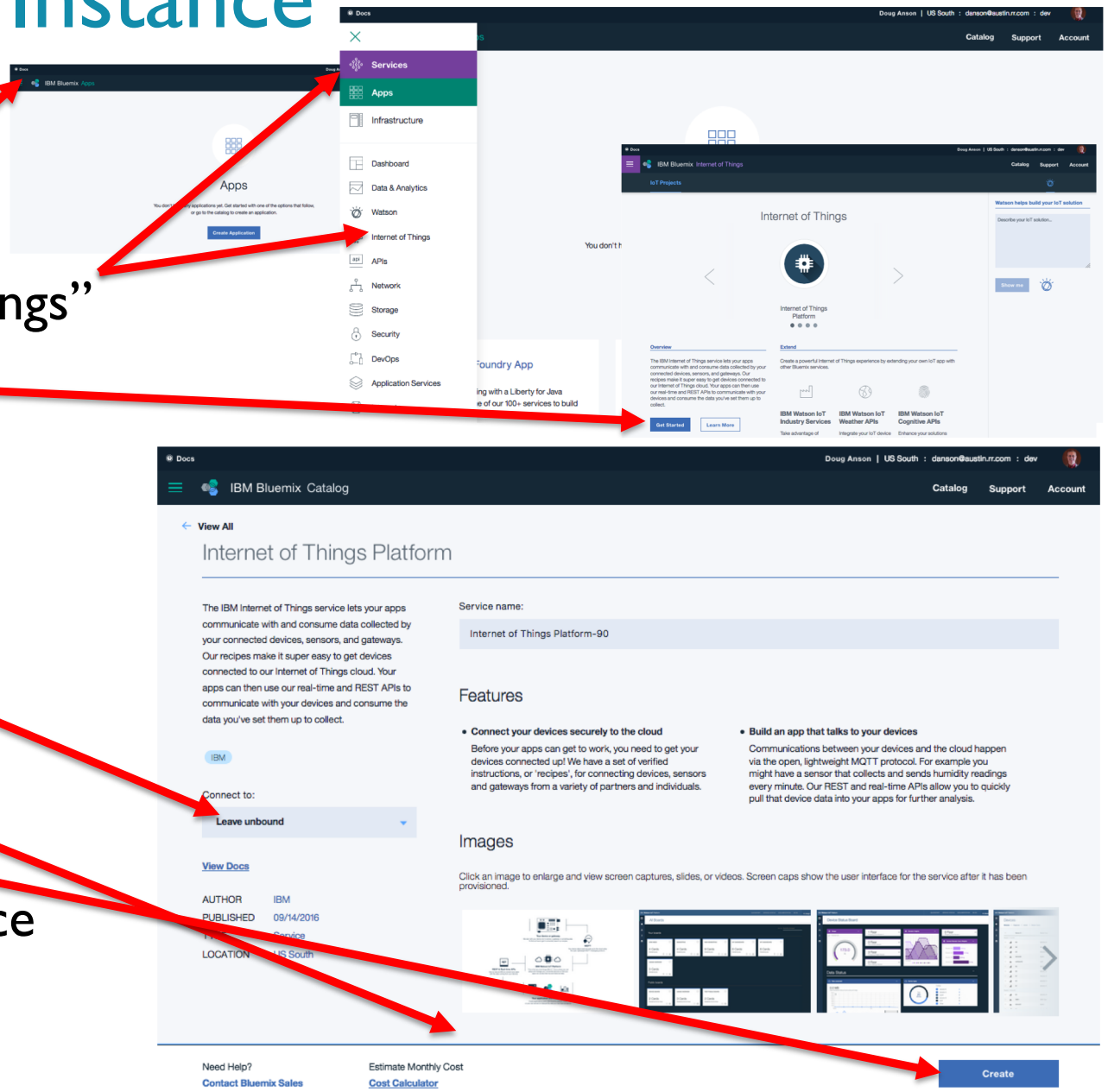
What we will do next…

- Create our own Watson IoT Instance within our Bluemix account
- Create our Watson IoT Application Credentials (used in the NodeRED application...)
- Create our sample Watson IoT NodeRED Application
- Bind the Watson NodeRED Application to our Watson IoT instance
- Create a mbed Cloud API Key/Token
- Configure the Watson IoT ARM mbed Cloud Bridge

Create our Watson IoT Instance

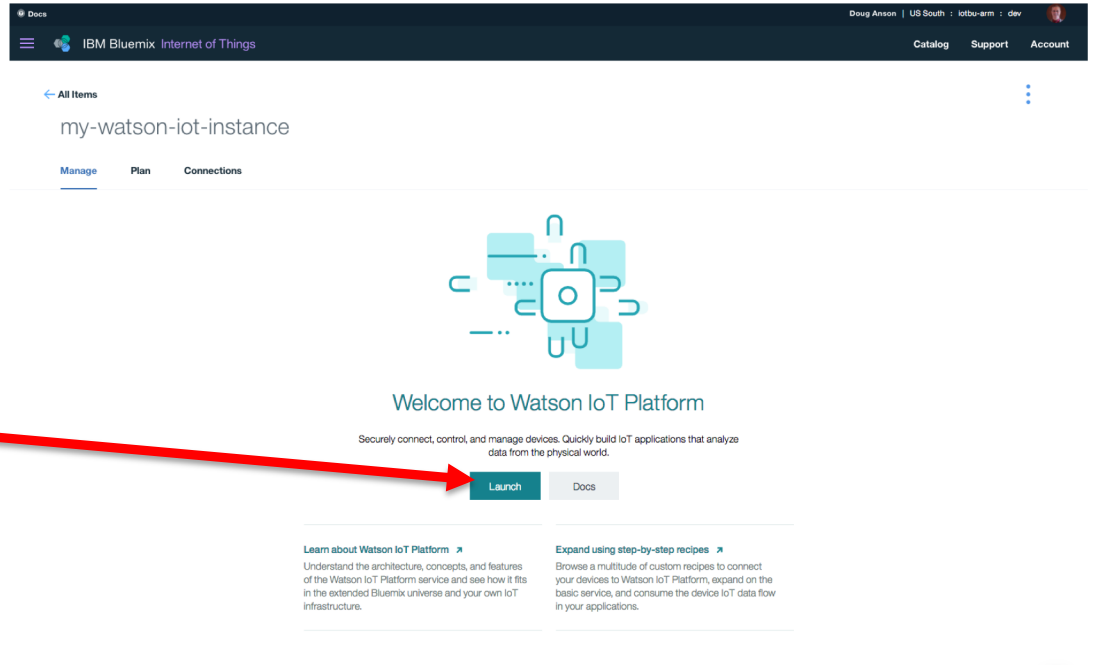
- Go to the Bluemix dashboard & click: (https://console.bluemix.net)
- Select "Services", then "Internet of Things"
- Press "Get Started"
- Configure your Watson IoT Instance
  - Leave "unbound"
  - Select the "Free" plan (scroll down…)
- Select "Create"

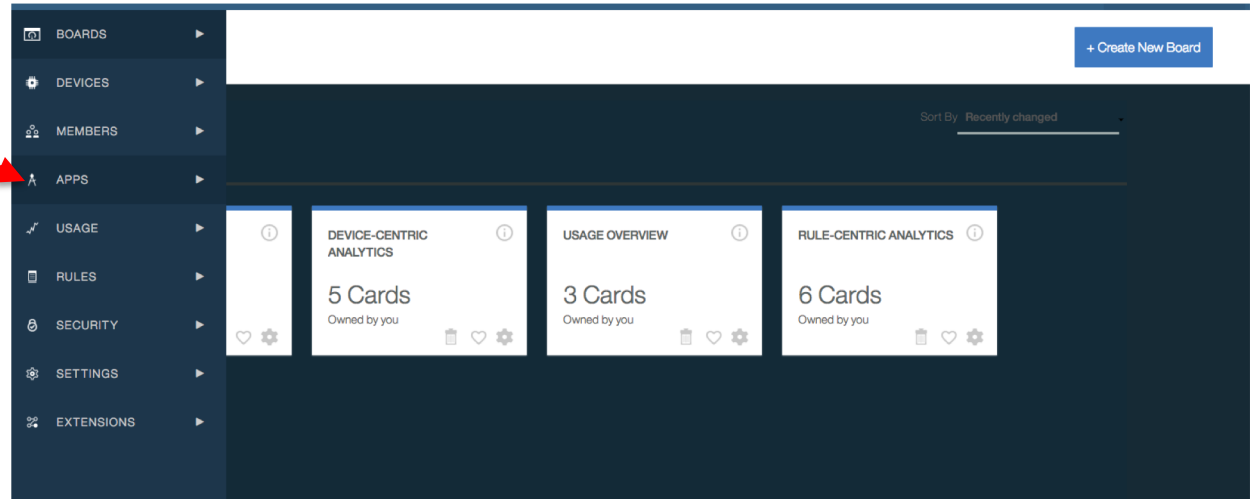
This will create your Watson IoT instance



Create Watson IoT Application Credentials

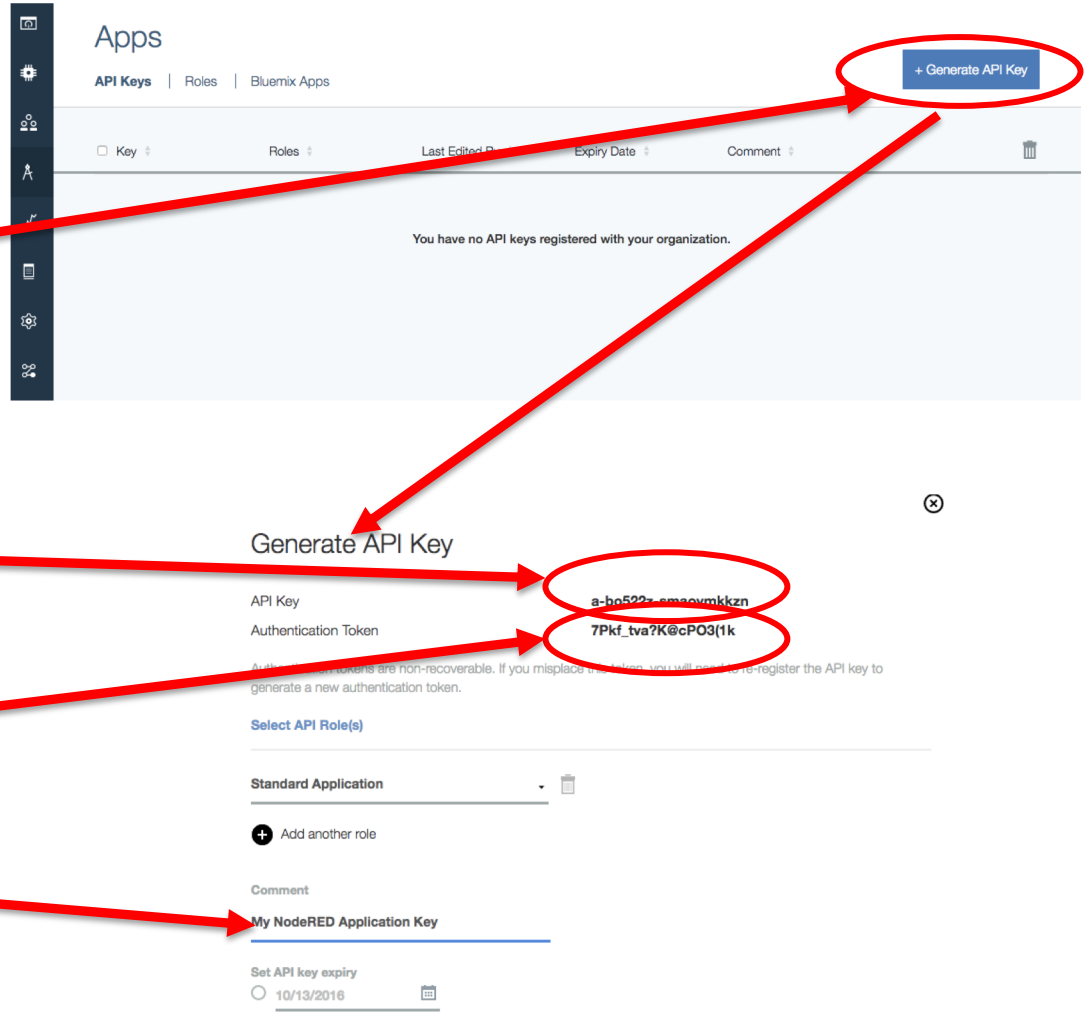
- Once your Watson IoT Instance is created, you should see the Watson IoT dashboard
- Launch the "Launch"
- Press "Apps"





Create Watson IoT Application Credentials…

- Press "Generate API Key"
- Record the API Key
- Record the Authentication Token
- Press "Generate" after adding a comment
- Save these two values! You will need them later…

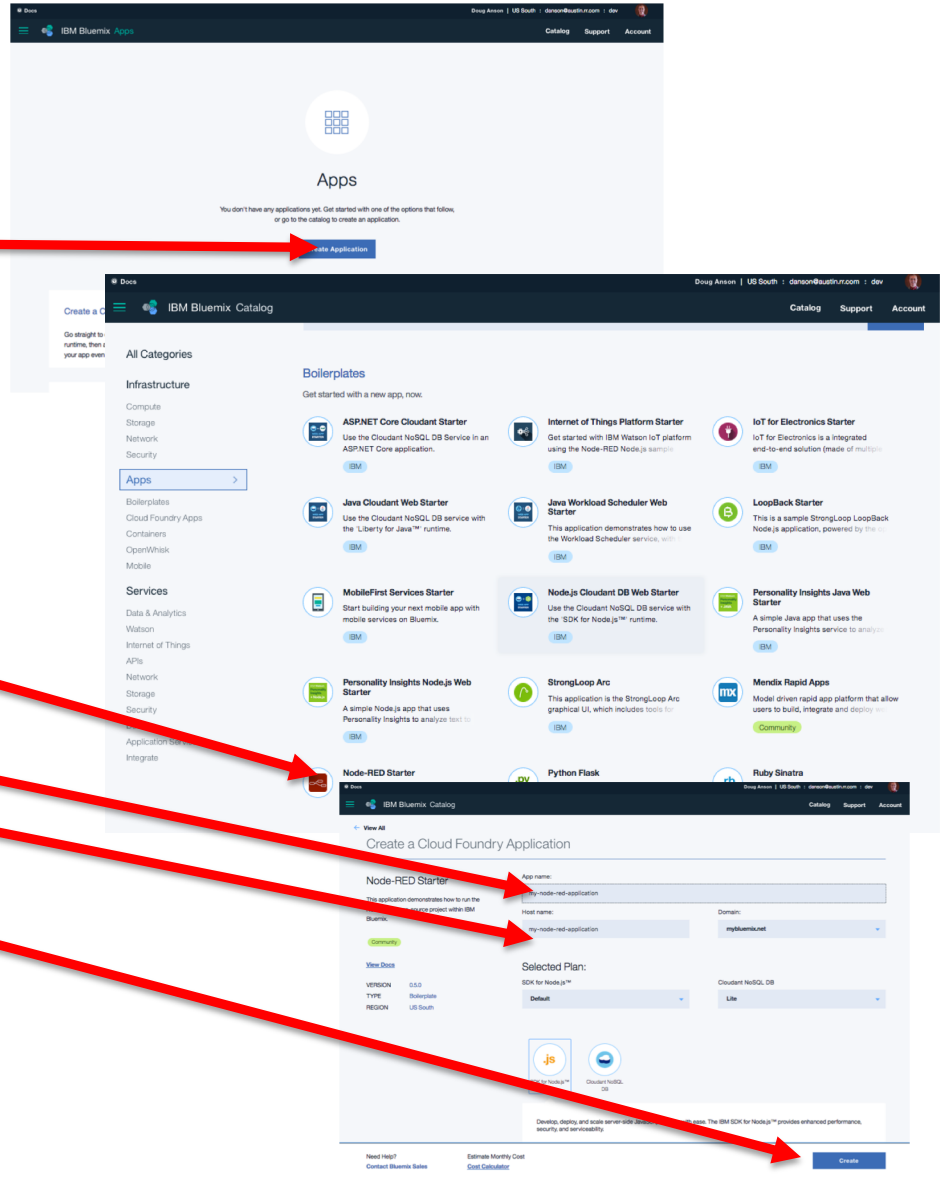


ARM

Creating our Sample Watson IoT NodeRED Application

Please DO NOT USE Internet Explorer for this task!

- Go back to our Bluemix Dashboard: https://console.bluemix.net
- Click on "Create Application"
- Select "NodeRED Starter"
- Enter an "App Name" (must be unique)
  - Make a note that your app URL is <app name>.mybluemix.net
  - Press "Create" to finalize creating the application
- Application will "Stage"… this will take a few minutes...
- Next we will configure the NodeRED application (next slide)

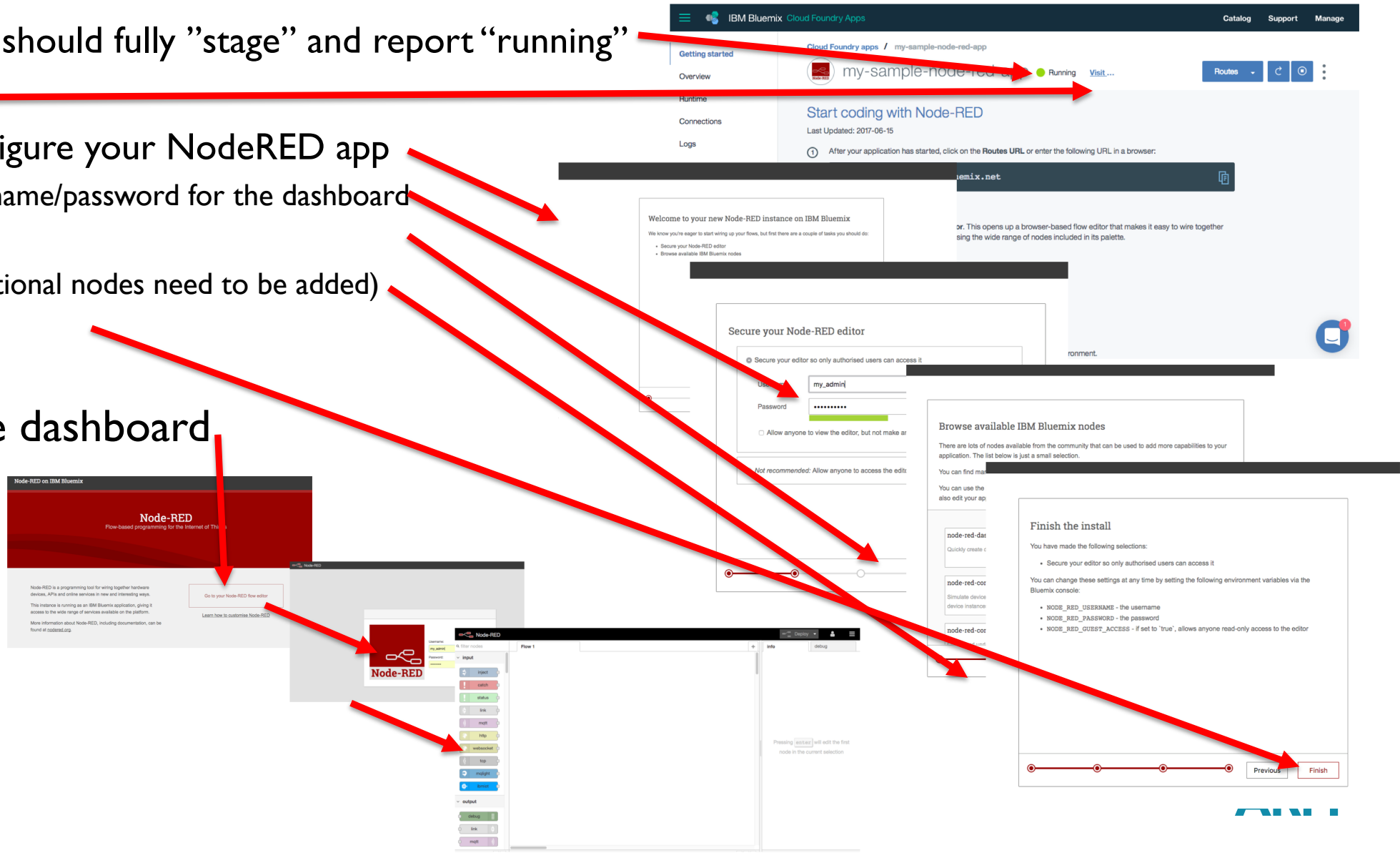


Configure Watson IoT NodeRED Application

Please DO NOT USE Internet Explorer for this task!

- Your NodeRED app should fully ”stage” and report “running”
- Click on “Visit...”
- Next, you must configure your NodeRED app
  - Create an admin username/password for the dashboard
  - Press “Next”
  - Press ”Next” (no additional nodes need to be added)
  - Press “Finish”

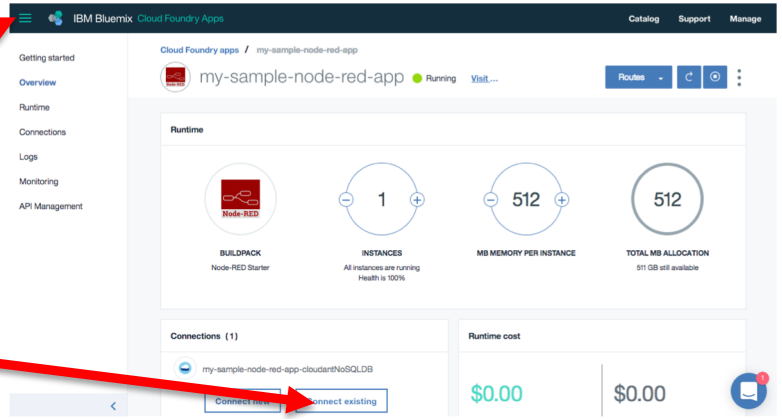
- Now, login to the dashboard



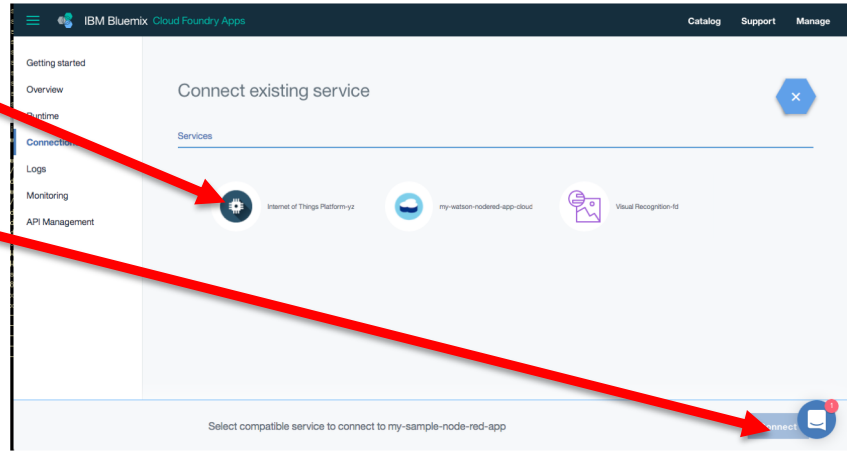
Bind the Watson NodeRED Application to our Watson IoT instance

Please DO NOT USE Internet Explorer for this task!

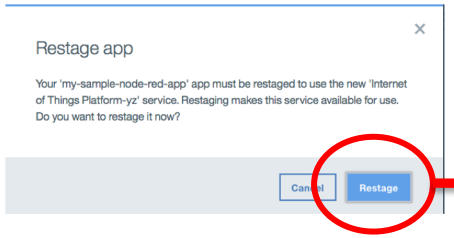
- Dashboard -> Apps, select your NodeRED app

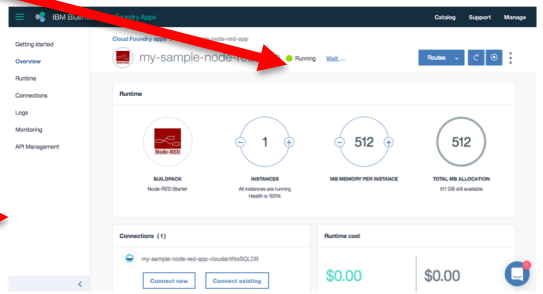


- Press "Connect Existing"
- Click on your Watson IoT Instance
- Press "Connect"



- Your NodeRED application will "restage"… this will take awhile. Wait for the "running" state:



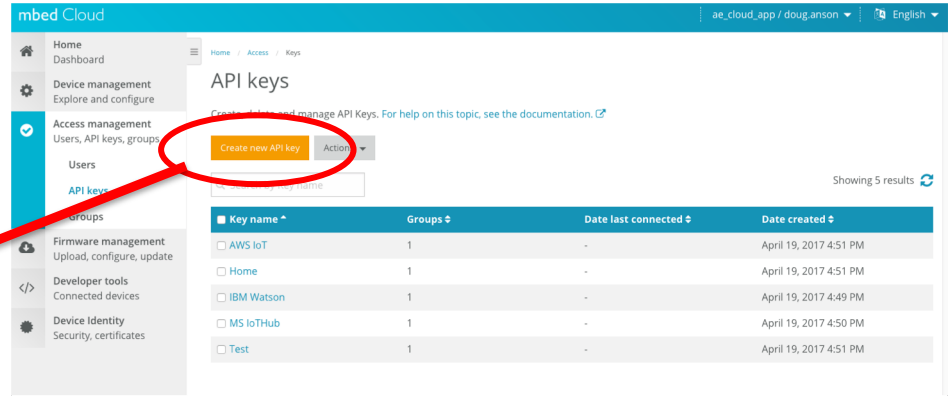


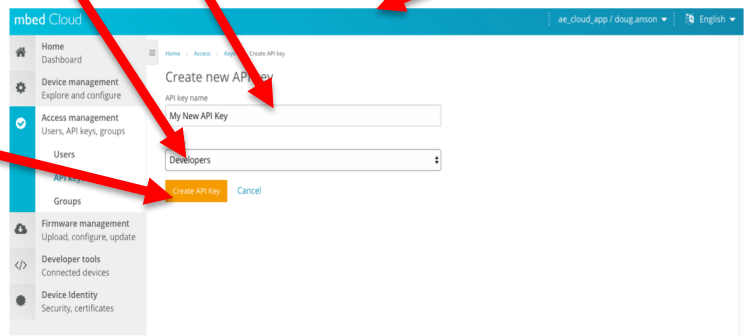
Note: keep this window open!!!

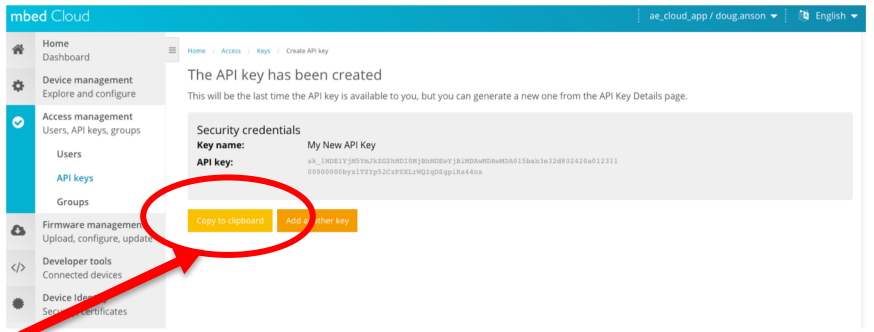
ARM

Create our mbed Cloud Access/API Token

- Navigate to the mbed Cloud Dashboard: https://portal.us-east-1.mbedcloud.com
- Log in, Select "Access Management", then "API keys"
- Press "Create new API Key "… you will create a key
- Give the new API Key a name
- Select "Developers" group
- Create the API Key
- Press "copy to clipboard" – save to a file for later use

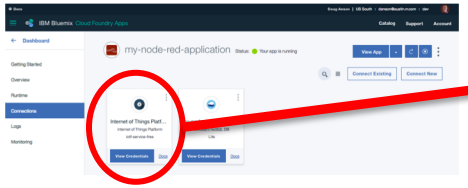


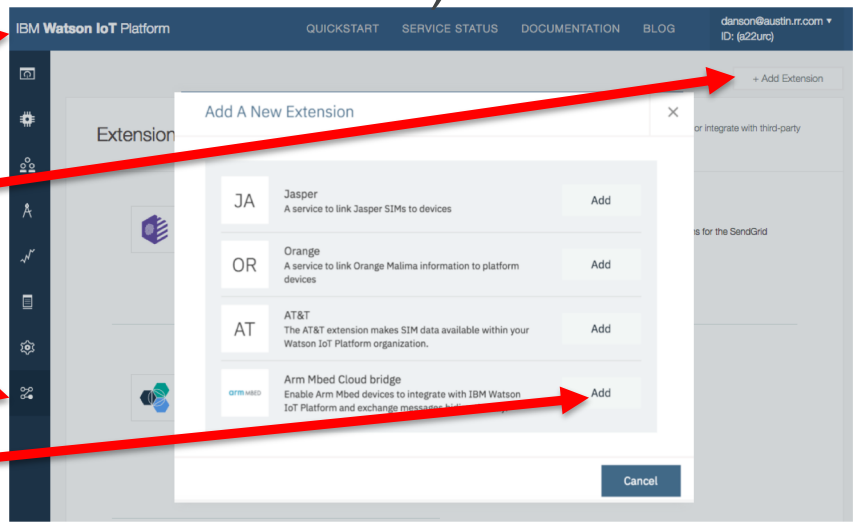




Configure the Watson IoT ARM mbed Cloud Bridge

- Go back to the Watson IoT Dashboard (last window open on slide 21…)

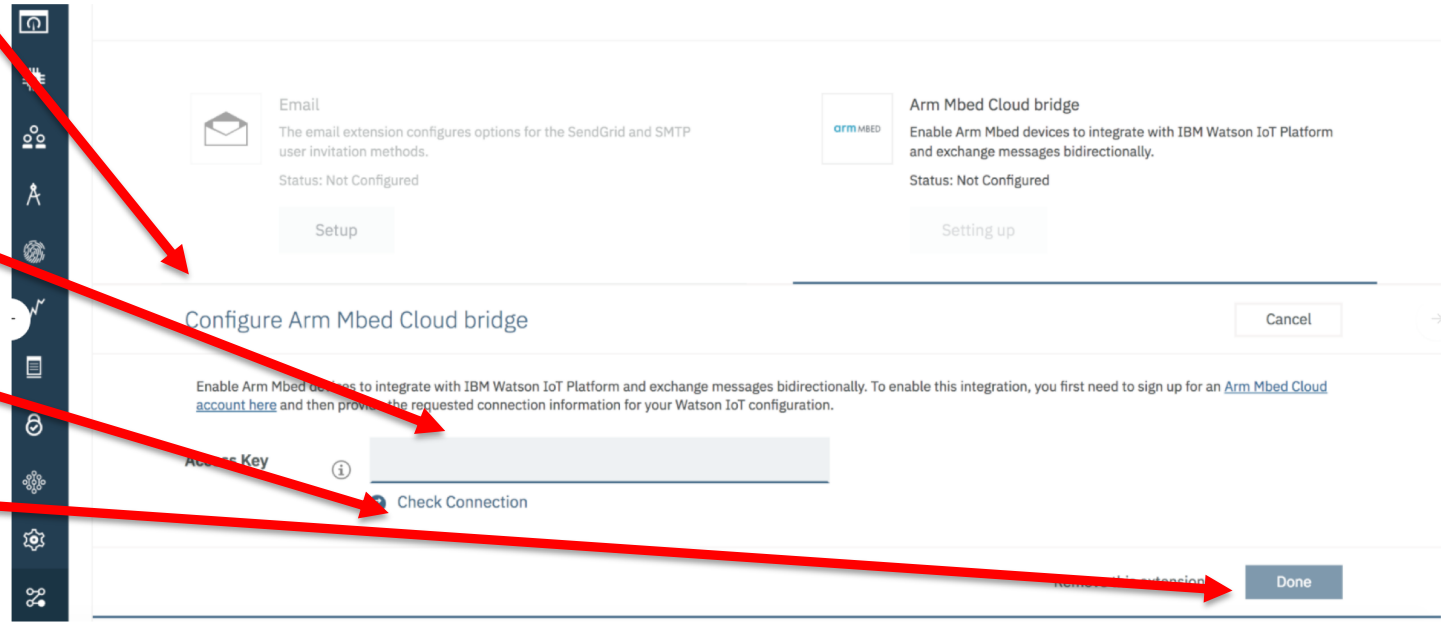




- Select "Extensions" (bottom icon)
  Select "Add Extension"

- Select "ARM mbed Cloud"
  (Press "add", then "setup")
  - scroll down a bit…

- Paste your mbed Cloud API Key
- Press "Check Connection"
- If OK, press "Done"



Status Check

So far we've completed the following

- Created our own Watson IoT Instance within our Bluemix account
- Created our Watson IoT Application Credentials (used in the NodeRED application...)
- Created our sample Watson IoT NodeRED Application
- Bound the Watson IoT Application to our Watson IoT instance/service
- Created a mbed Cloud API Token
- Configured the Watson IoT ARM mbed Cloud Bridge

Next, we will Import our NodeRED flow sample and restart the Endpoint... We should then see device telemetry flowing from the device, through mbed Cloud, through the Bridge, and into Watson IoT!

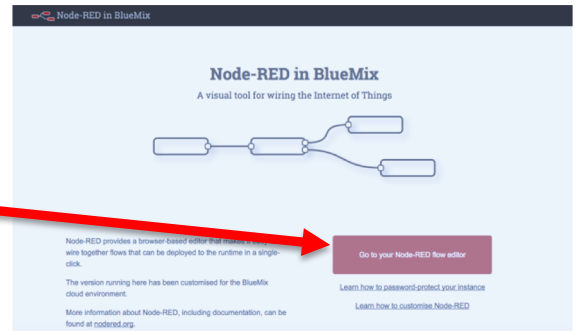
Importing the NodeRED Flow Example

Copy the Sample NodeRED Flow JSON

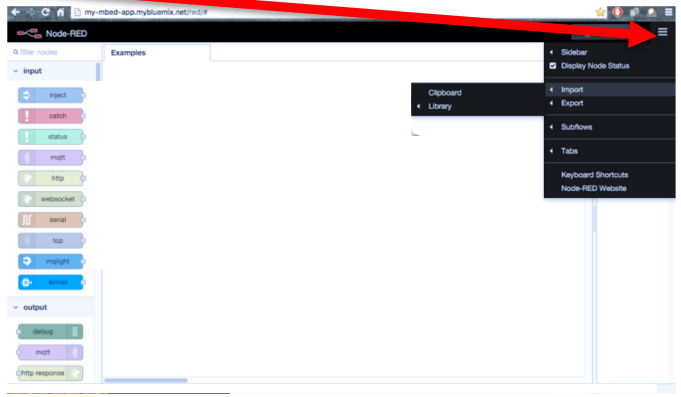
- Go back to your Online IDE workspace
- Go to your "mbed-cloud-sample" project
- Double-click on NODEFLOW.txt
- Copy all of that file…

Import the NodeRED Flow

- Navigate to the URL that we recorded earlier for your Watson IoT NodeRED Application (i.e. http://<app name>.mybluemix.net)
- Select "go to your NodeRED flow editor"



- Select the menu (far right)
- Select "Import"
- Select "Clipboard"

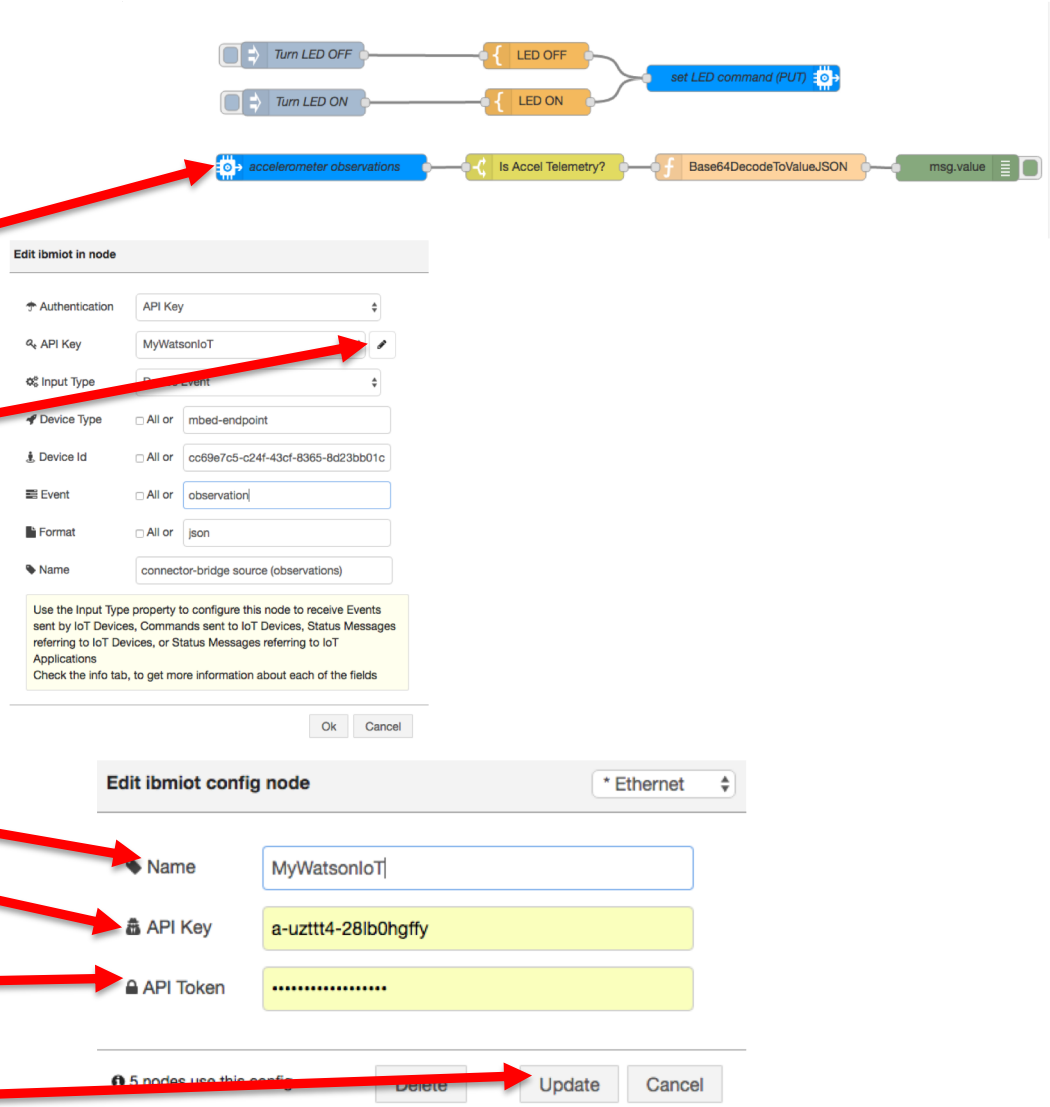


- Paste the entire contents of the JSON code you copied in the previous slide into the "paste nodes here" window… then press "OK".
  - If you have trouble copy/pasting the NODEFLOW.txt file contents, try copying it from https://github.com/ARMmbed/mbed-cloud-sample (NODEFLOW.txt is listed there…click & copy…)

ARM

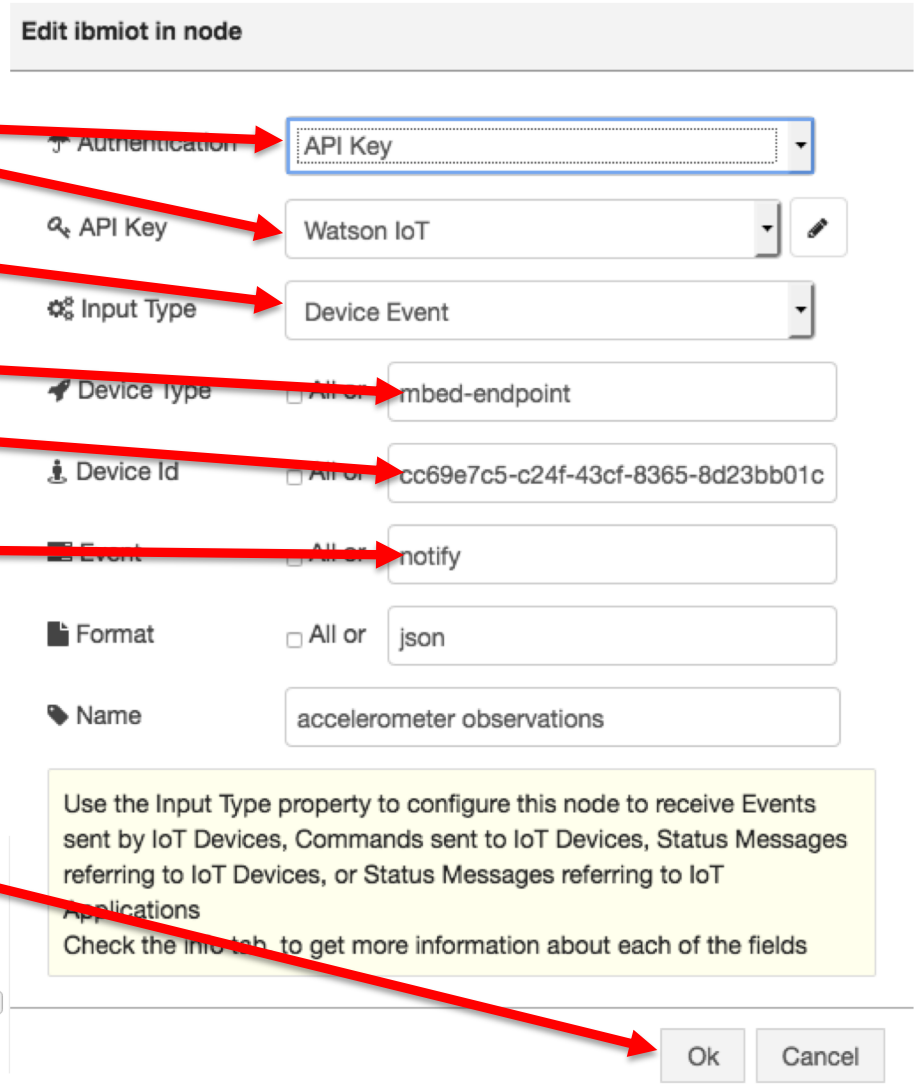
Your new Watson IoT Application Node Flow: Configure it

- Configure your NodeRED flow input and output nodes (Blue) and link them to your Watson IoT instance
- Click on the observation node
- Click on the "edit" button
- Provide a name
- Insert your Watson API Key from slide 20
- Insert your Watson Auth Token from slide 20
- Press "Update" to save

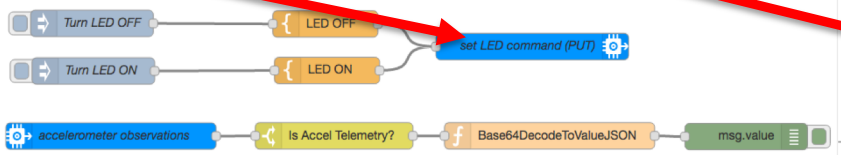


Your new Watson IoT Application Node Flow: Configure it…

- Now, select the new linkage to your Watson IoT instance via the drop down here and that "API Key" is selected
- Input Type is "Device Event"
- You can filter by Device Type (found in main.cpp on your endpoint) or "all"
- Ensure your have your endpoint ID here… this comes from mbed Cloud Dashboard->Developer Tools->DeviceName
- Event is "notify" (i.e. CoAP observations)
- Press "OK"



- Edit the blue PUT node
  - API Key Auth
  - Select same API Key
  - Device Type, Device ID
  - Event is "PUT" (all caps!)
  - Input Type is "Device Command"

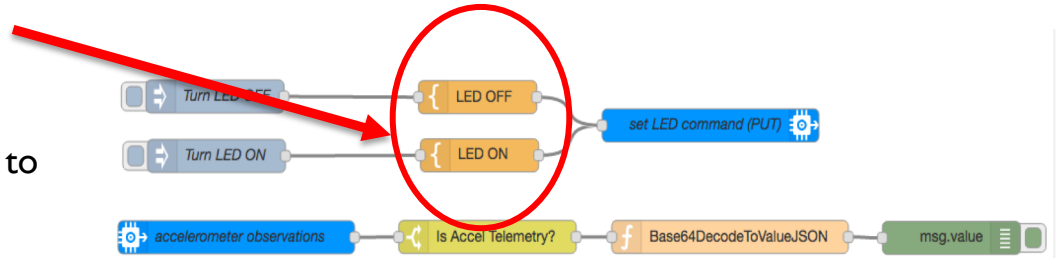


ARM

Your new Watson IoT Application Node Flow: Configure it…

- Lastly, we have to look at the remaining "Orange" command builder nodes and update them as well

- For both LED OFF/ON nodes, select and note the JSON payload created. "deviceId" needs to be the value you have for your Device Name (from mbed Cloud Dashboard→ Developer Tools)
  - "deviceId" will be on the right-hand side in the JSON structure… you have to scroll to the right to see it... You can also enlarge the edit window…



- Locate MBED_ENDPOINT_NAME_GOES_HERE and replace it with your actual Device Name value … press "Done" when complete.

- FYI, Each LED ON/OFF has a Base64 encoded value
  - It's a "1" for ON, "0" for OFF… each must be Base64 encoded

- Press the red "Deploy" button in the upper right hand side of your NodeRED console

ARM

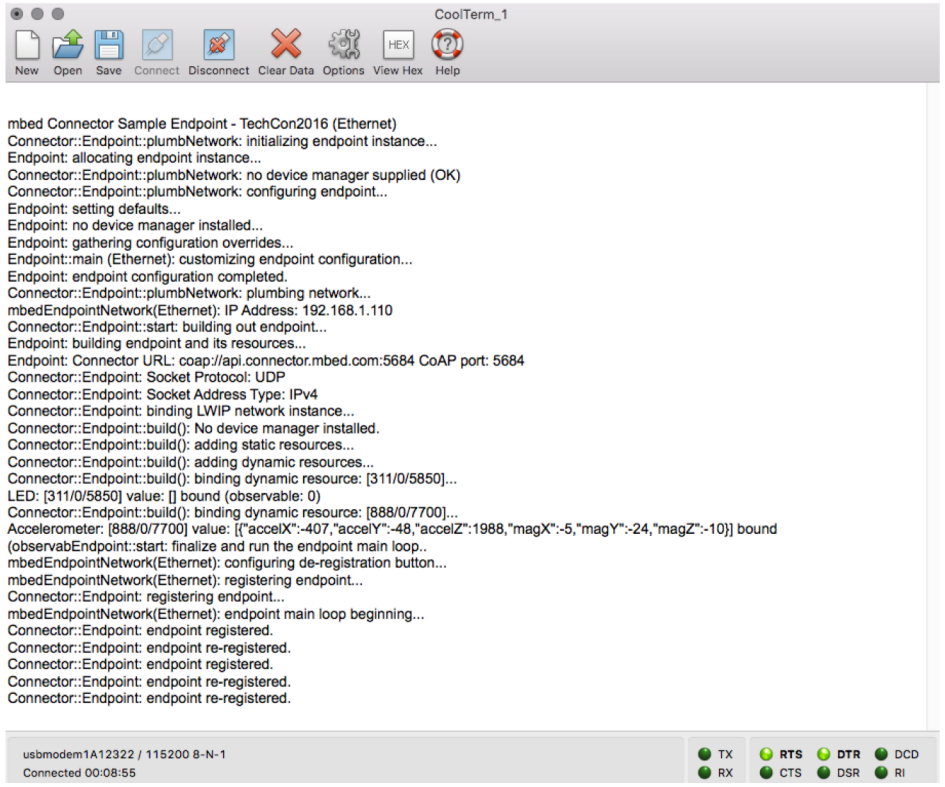
Putting it all Together

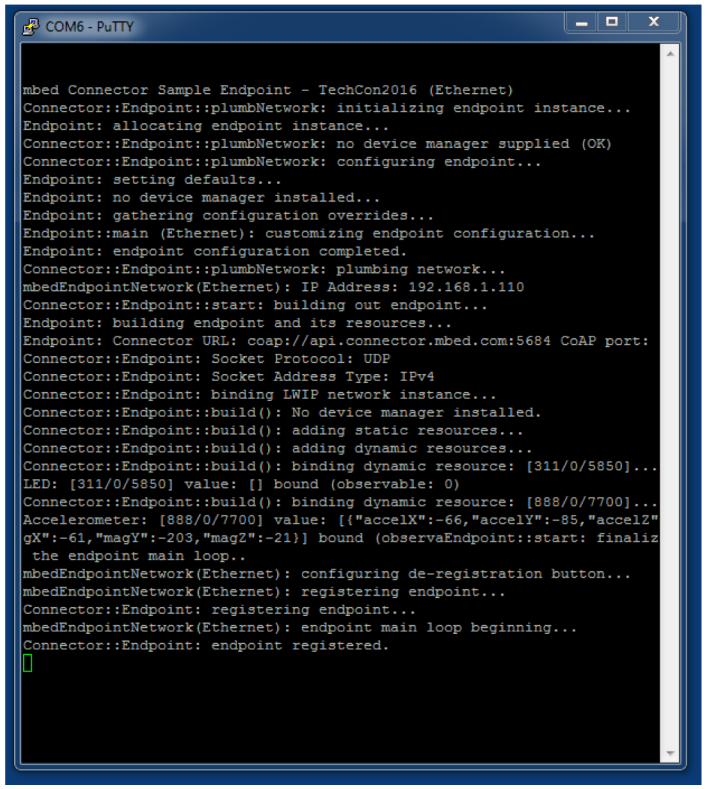
- Press the "reset" button on your mbed device (next to the USB port)
- On your NodeRED Editor, select the "debug" window

Lets check how things are working…

Putting it all Together: Check the Serial Terminal

- You should see output that looks something like this:
  - Make sure that you see a message like "endpoint registered"





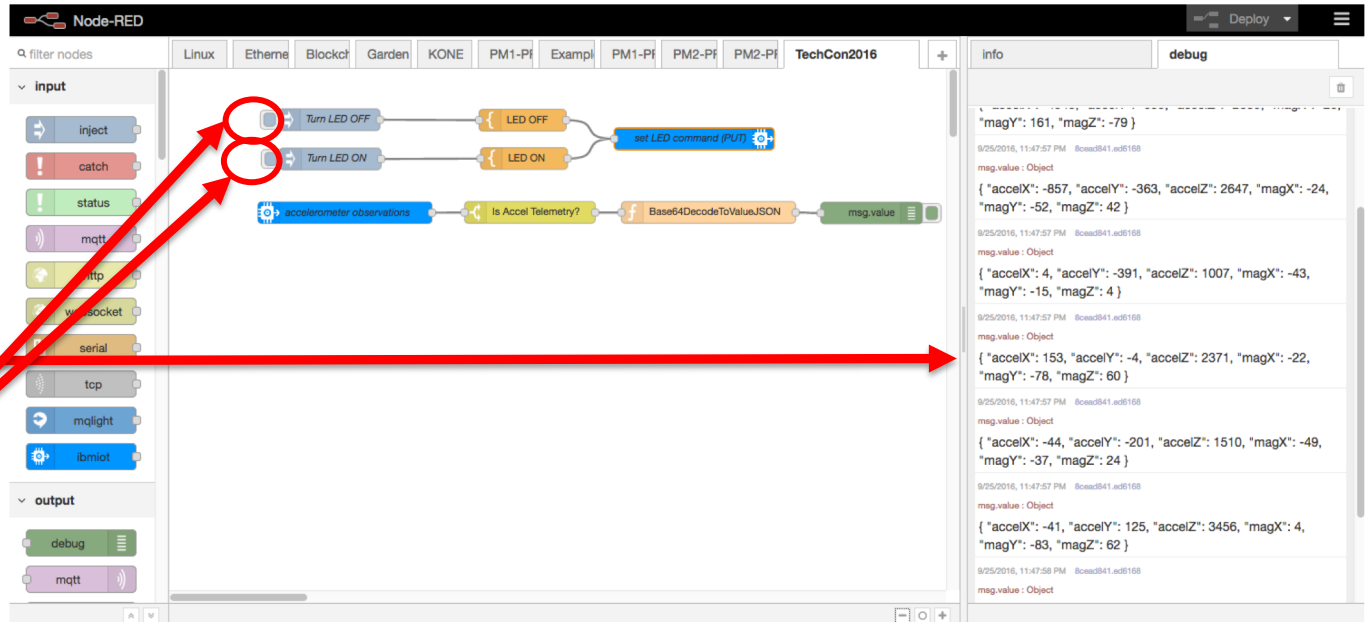
Putting it all Together: NodeRED debug info…

- Navigate to your Watson IoT application and NodeRED flow editor

- Examine the debug window
  - Shake your K64F!

- You should see output similar to this… telemetry in Watson IoT

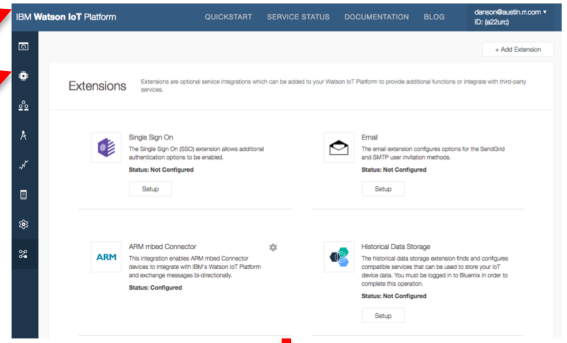
- You can toggle your LED On and off (left-most block) by clicking each of these… look in the serial terminal for output from the endpoint



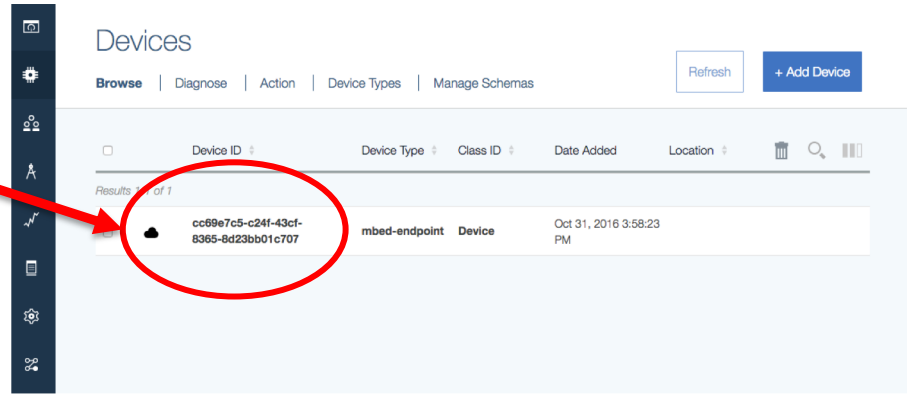
ARM

Putting it all Together: Check Watson IoT Devices

- Go to the last window you had open from slide 19
- Select ”Devices”



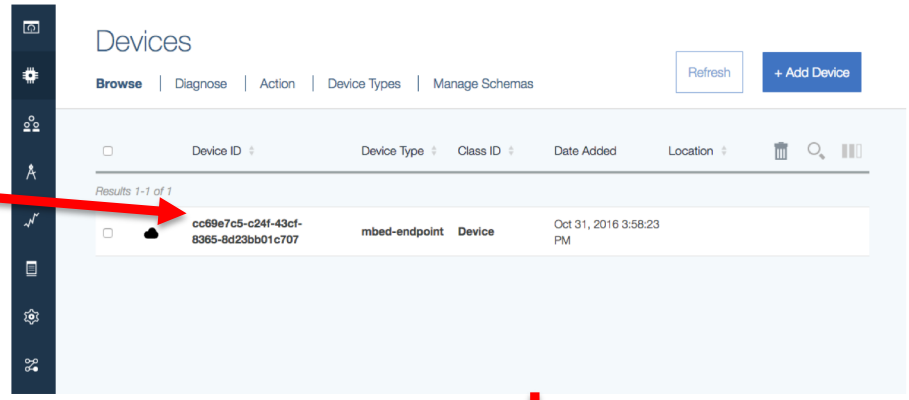
- The bridge automatically creates Watson IoT devices for you



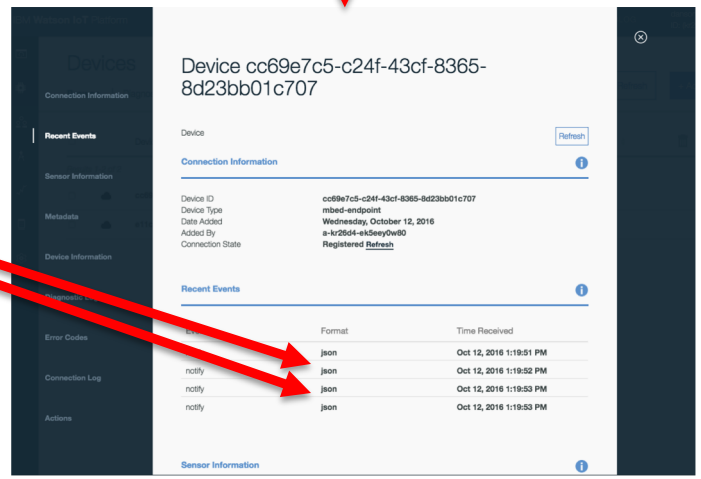
ARM

Putting it all Together: Check Watson IoT Devices

- Select your device



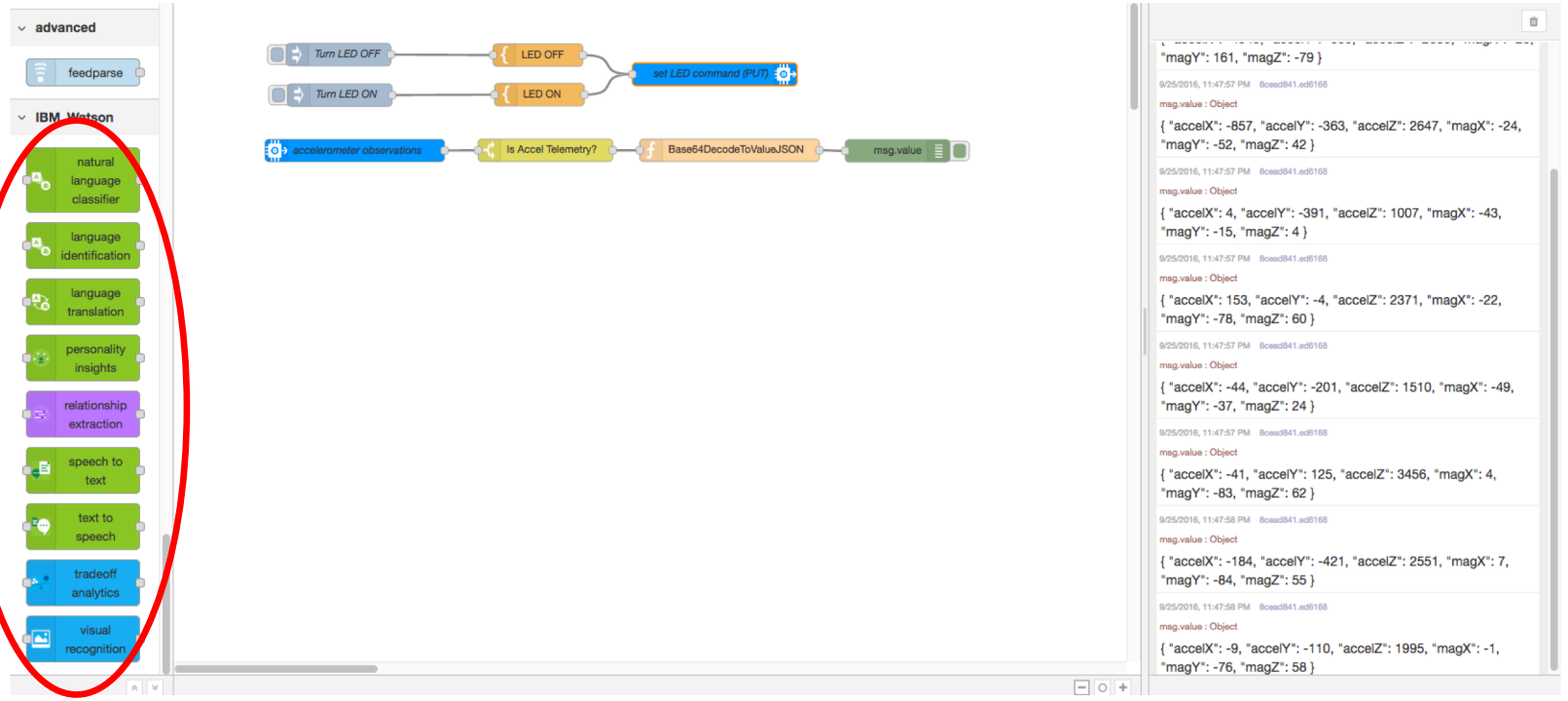
- You can watch CoAP notify events occurring via your bridge (observations)



ARM

Ah Ha!

This is SUPER COOL



- CONGRATS! You have now finished getting mbed device data into IBM Watson IoT.
- We can deliver mbed device data into Watson IoT analytics via our NodeRED flow!
- The bigger challenge remains... What can/do you do with your data telemetry?

ARM

Summary

We have completed the following – congratulations!

- Created our Bluemix mbed environments and PC tool setup
- Imported our mbed endpoint, customized, installed, and ran it
- Created our own Watson IoT Service Instance and NodeRED application
- Configured our Watson mbed Cloud Bridge
- Imported a NodeRED flow and customized it
- Examined live telemetry and some bi-directional capabilities of our bridge

Great JOB! Thanks for your time.