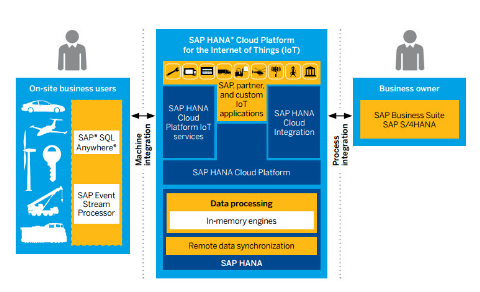
IoT1C09 – Setup a Tessel Device

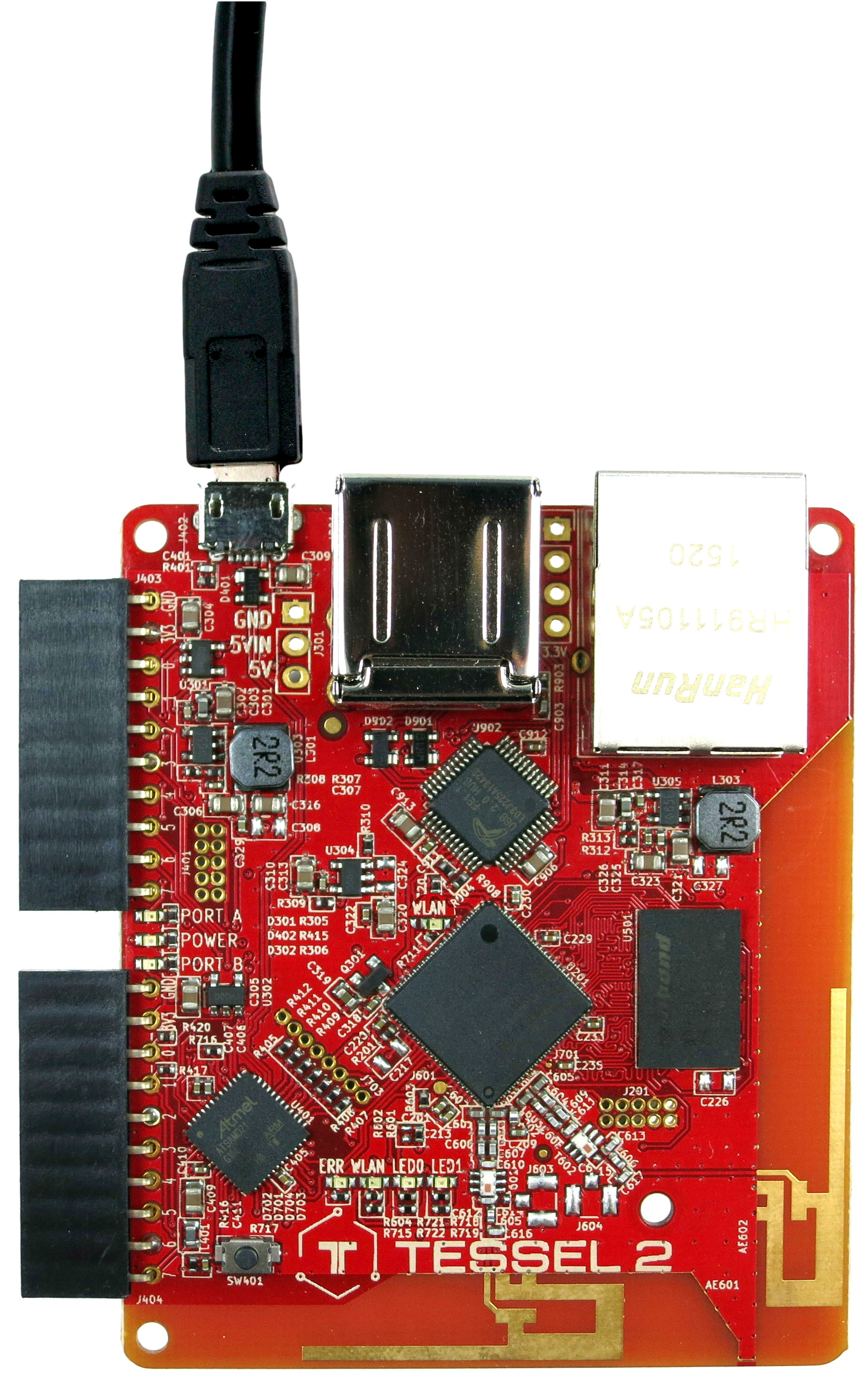
|  |  |
| --- | --- |
| **Product**  HANA Cloud Platform IoT Services  **Level**  Undergraduate/Graduate  Beginner  **Focus**  HANA Cloud Platform  **Author** Ross Hightower | MOTIVATION  In this case you will configure a Tessel device to feed data into your HANA services.  **PREREQUISITES**  None |



# 

# Setup a Tessel Device

A Tessel is a small computer to which you can attach many sending devices. This case uses the climate module which records temperature and humidity but there are many more available.



The nice thing about a Tessel is that you can write programs for it using Node.js using JavaScript. This makes it relatively easy to use.

You can find a lot of information at this site.

https://tessel.io/

## Setup the Tessel

This case uses a Tessel 2 and the instructions for setting up the device can be found here:

<http://tessel.github.io/t2-start/index.html>

At the time this case was written, the Tessel software only support node.js version 4. If you already have a later version of node.js you can use one of the following to manage multiple versions:

Windows nvm-windows <https://github.com/coreybutler/nvm-windows>

OS-X nvm <https://github.com/creationix/nvm>

This case uses the climate module for the Tessel so complete the setup for the Tessel and then continue to test the climate module.

## Create the Application

The code shown below was adapted from the Tessel code found at this site:

<https://github.com/saphanaacademy/IoTServices>

The only modifications is the addition of the timestamp field in the message and the interval for sending messages to HCP has been changed to 10 seconds instead of 1 second.

If you ran the test program for the Tessel climate device you can create the new file in the same folder. If you didn’t run the test program yet, go back to the Tessel setup and complete that step.

Create a new file called **climateOnPremise.js** and copy the code below into it.

|  |
| --- |
| var tessel = require('tessel');  var climatelib = require('climate-si7020');  var climate = climatelib.use(tessel.port['A']);  // Connect to SAP HANA  var http = require('http');  var httpOptions = {  host: '<host>,  port: <port>,  path: '<path>', method: 'POST',  headers: {  'Authorization': <Authorization Header>',  'Content-Type': 'application/json'  }  };  climate.on('ready', function () {  console.log('Connected to climate module');  // Loop forever  setImmediate(function loop () {  climate.readTemperature('f', function (err, temp) {  climate.readHumidity(function (err, humid) {  console.log('Degrees:', temp.toFixed(4) + 'F', 'Humidity:', humid.toFixed(4) + '%RH');  updateIoT(temp.toFixed(4), humid.toFixed(4));  setTimeout(loop, 3000);  });  });  });  });  climate.on('error', function(err) {  console.log('error connecting module', err);  });  // Generate JSON output and send over POST request to HANA  // Brightness field added if the ambient module is used in the future  function updateIoT(temp,humid) {  // catch HTTP request errors  var req = http.request(httpOptions).on('error', function(err){ console.error(err); });  var jsonData = {  "ID": "1",  "BRIGHTNESS": "27",  "TEMPERATURE": temp,  "HUMIDITY": humid,  };  var strData = JSON.stringify(jsonData);  console.log(strData);  req.write(strData);  req.end();  }; |

Listing

Fill in your HCP parameters at the highlighted parts of the code near the top.

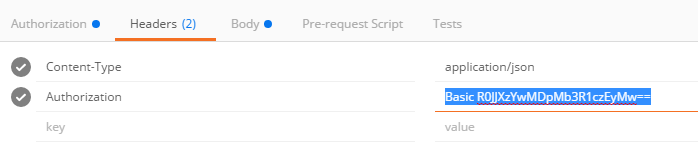
Get the host and port from the URL for the WDW you received from you UCC.

Host: For example: hd3.hana.ucc.uwm.edu

Port: For example 8003

Path: this the part of the URL after the port when you run your .xsodata file. For example: /GBI\_600/iot/iot.xsodata/DATA

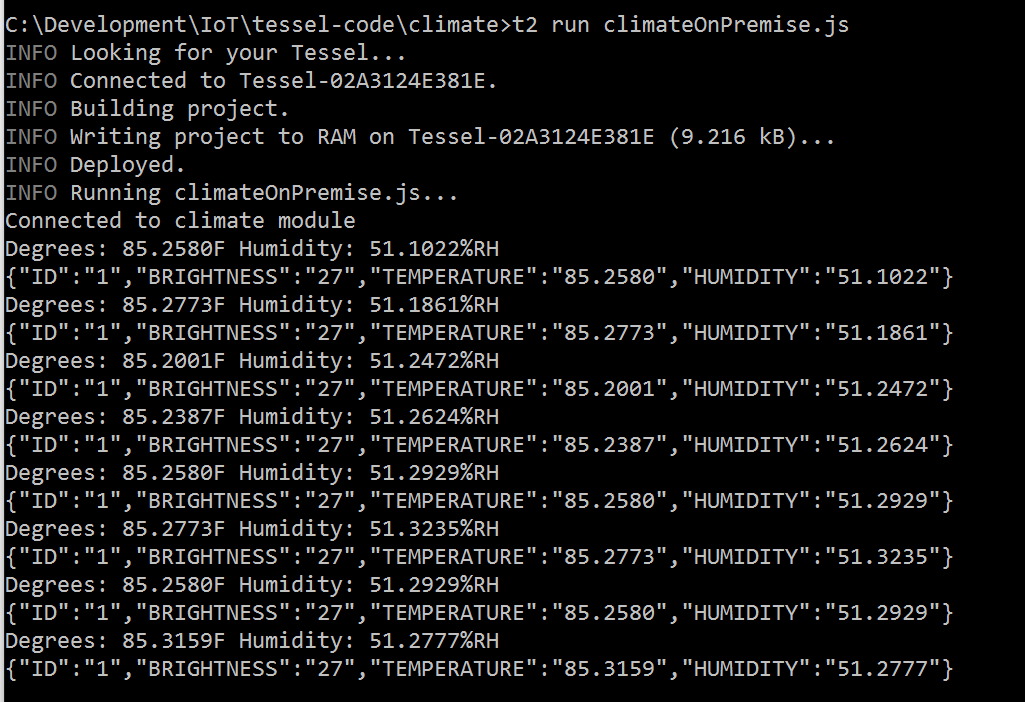
To get the Authorization Header, open Postman and select the POST message you used to test the service in a previous case. When you entered your user id and password and clicked Update Request, Postman created an Authorization header. Copy the header value and paste it in the appropriate place in the program.



Save the file then run it on the Tessel from a command prompt using the following command:

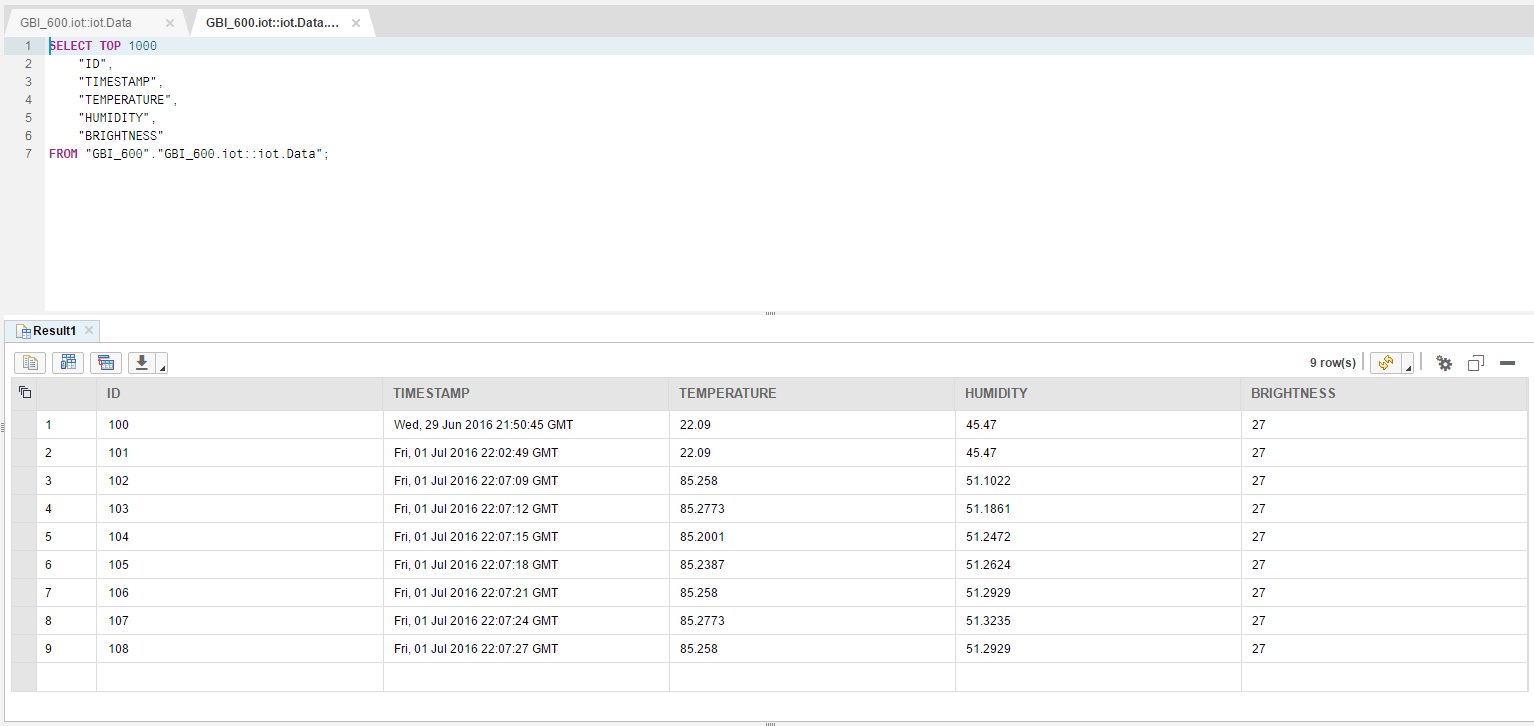
**t2 run climateOnPremise.js**

The Tessel will begin sending messages every 3 seconds.



## Check the Data

You can check the data in two ways. You can open the Catalog editor and open the iot.Data table. Click the Open Content button to see the data.



The other option is to use the OData service you created. Open the .xsodata file in the editor and click run. That opens the service document. Add /HISTORY to the URL to see the data in XML format.

