Programming Assignment: End-to-end scenario

You have not submitted. You must earn 1/1 points to pass.

Deadline Pass this assignment by May 21, 11:59 PM PDT

Assignment goals

This assignment will create a NodeRED application on both the Raspberry Pi and on Bluemix to create an end-to-end scenario. You will send temperature data from the Raspberry Pi and SenseHAT to a cloud based application where it is analysed. The cloud application will send a command back to the device if the temperature rises above, or falls below a pre-set limit. The Raspberry Pi will display the current temperature status, based on the latest cloud application command and also the current temperature from the SenseHAT temperature sensor.

Part 1 - Register your Raspberry Pi as a gateway

This assignment will use the Raspberry Pi as a gateway, publishing data on behalf of a sensor and also publishing gateway data.

To accomplish this you will:

- delete the registration of the Pi in your Watson IoT Platform organization,
- create a new gateway type for your raspberry Pi then register your Pi as a gateway
- use the SenseHAT as the remote sensor, so need to create a device type and register the senseHAT

Part 2 - Publish temperature data

Publish the senseHAT temperature and humidity every 5 seconds from the SenseHAT device via the Raspberry Pi Gateway.

The CPU temperature of the Raspberry Pi should also be published to the Watson IoT organization every 5 seconds, but published from the gateway, not the sensor device.

See the "How to complete the assignment" section for the correct format of the command payload.

Part 3 - Bluemix application to respond to the temperature data

Create a Bluemix application to:

- receive the SenseHAT data
- check to see if the senseHAT device data is above or below a specified temperature (29ºC). If the temperature rises above the specified temperature then send a command to the device to turn a warning LED on, if the temperature is below the specified temperature then send a command to turn the warning LED off. The command should only be sent when the temperature rises to or falls below the specified temperature - not every time data is received from the device.
- See the "How to complete the assignment" section for the correct format of the command payload.
- To help debug your code you may find it useful to output the temperature data from the gateways and the sensors to the debug tab in Bluemix UI.

Part 4 - Update the SenseHAT LED panel

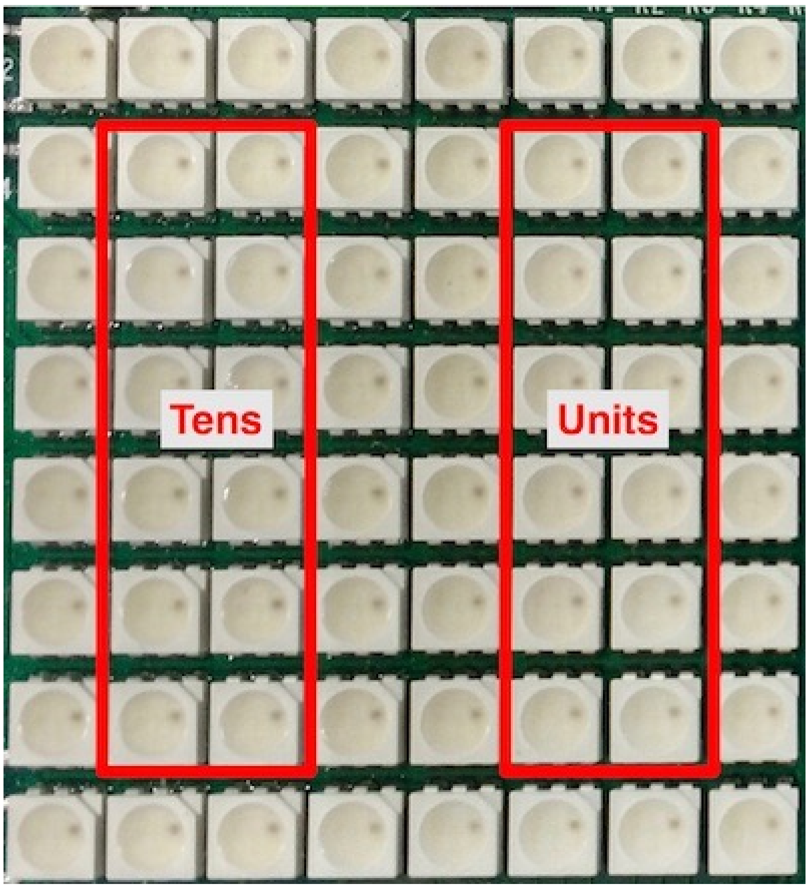
On the device update the NodeRED application to receive the command. The LED panel on the SenseHAT should then be used to display appropriate data:

- If the command to turn the warning LED on is received the entire display should be set to maroon (#800000) RGB(128, 0, 0)
- If the command to turn the warning LED off is received the entire display should be set to green (#008000) RGB(0, 128, 0)
- if no command has yet been received the LED panel should be black (off / #000000) RGB(0, 0, 0)

In addition to the color being set above the LED panel should also display the current temperature being reported by the senseHAT by overlaying the temperature data over the background colour:

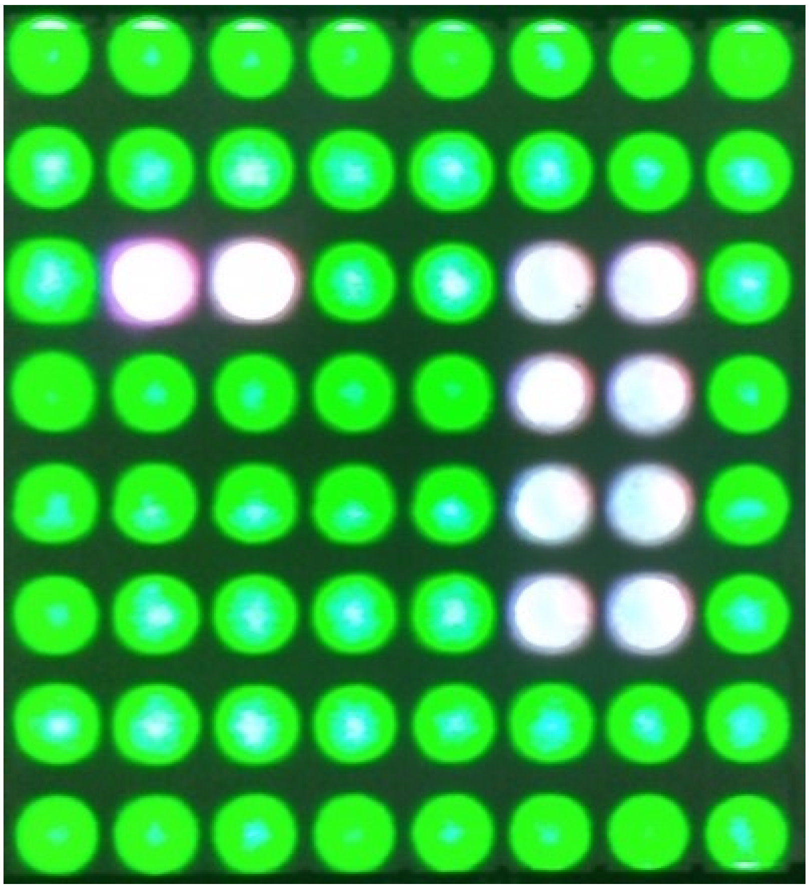
- take the temperature and round down to nearest integer (29.9 = 29)

- 10s part of the temperature should be displayed in columns 2 and 3 of the LED panel in rows 2-7
- units part of the temperature should be displayed in columns 6 and 7 of the LED panel in rows 2-7
- Represent the value by turning the appropriate number of LEDs to silver (#C0C0C0) or RGB(192,192,192). If the value for a section is 0 then no LEDs should be silver, so for 30 there should be 3 LEDs in the Tens section showing silver but no LEDs in the Units section showing silver.

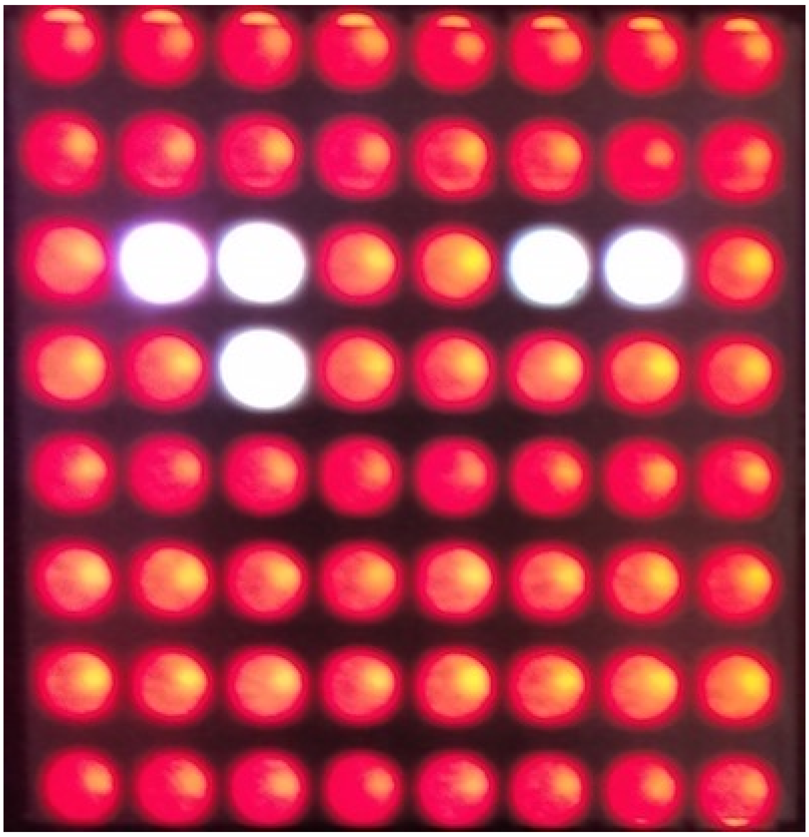


NOTE: column 2 is the second column, but its coordinate is 1 as the grid starts at 0. The same applies to rows.

28°C =



32ºC =



- You can light any of the LEDs in each region, so long as the correct number are set to silver to represent the value. Those LEDs not silver should be the appropriate color as defined above for when a command is received

NOTE:

- **To update the LED values your solution should use the last command received (if one has been received) AND the latest temperature value from the SenseHAT.**
- **Your solution should combine the command string to update the background colour AND the temperature pixels into a single message sent to the SenseHAT output node.**
- **The LED panel should be updated every time the SenseHAT publishes a new temperature AND every time a new command to set the background is received.**
- **Only setting the background colour when a command is received and only setting the temperature when the senseHAT sends a new reading is not a valid solution.**

How to complete the assignment

SenseHAT deviceID = Raspberry Pi ID with 'sen' prepended.

e.g. If the Raspberry Pi is registered as a gateway with id 'a1b2c3d4' then you should register the SenseHAT as a device with id 'sena1b2c3d4'

Command Format

The command sent to the device should :

- have a command id/type of ‘display’
- be formatted as a JSON object
- have a single property called ‘screen’ with a value of either ‘on’ or ‘off’.

For example: **{"screen":"on"}** or **{"screen" : "off"}**

It must not include the temperature value nor have any additional content.

Note: If your code does not using the correct format for commands then you solution will not pass the grader and you will receive and error message saying your background, usually pixel (0, 0), is black RGB(0, 0, 0) rather than maroon (128, 0, 0) or green(0, 128, 0) as your code will not be responding to the correctly formatted commands used by the grader.

Event formats

Publish the CPU temperature data using the message format

{"d" : {"temp" : <CPU temperature>}}

The value returned from the vcgencmd should replace <CPU temperature> and should be a number not a string in the object published to the Watson IoT Cloud.

Publish the SenseHAT temperature and humidity data using the message format

{"d":{"temperature":<SenseHat temperature>,"humidity":<SenseHat humidity>}}

with the appropriate values received from the SenseHat. The values should be a number and not a string. As the SenseHAT sends a message every second, you must limit the rate of messages sent the the IoT platform to once every 5 seconds. This can be accomplished using the Delay node set to its rate-limiting mode.

Updating the LED panel

The display must be updated whenever a new sensor reading is received from the SenseHAT, or when a command is received from Watson IoT Cloud. The update must combine the most recently received command and temperature as described above into a single message sent to the SenseHAT.

What to submit

When you have the flows completed you should select all the nodes on the Raspberry Pi then export the flow to your clipboard. Open a text editor and copy the flow then save as a text file named assignment5.txt. Submit the text file containing your flow

Checklist

Here is a checklist to help you avoid the most common pitfalls experienced by previous learners:

- Are you using the correct command message when sending commands to the Pi? {"screen":"on"} or {"screen":"off"} are the only 2 valid commands. {"d":{"screen":"on"}} is not a valid message and values ON, On, OFF and Off are not valid values.
- Are you using the correct colours? Maroon, Green, Black and Silver are the only colours you should set in this assignment
- Are you updating the SenseHAT in a single message? Sending multiple messages to build up the display results in the incorrect display being shown for a small amount of time - may look OK, but is not a valid solution
- Are you updating the SenseHAT LED display every time a new message is received? (Waiting for the next temperature update to update the LED panel is not a valid solution)
- Are you updating the SenseHAT LED display every time a new temperature is released by the SenseHAT input node? (not every 5 seconds - every time a new

temperature is released. Waiting for the next command to update the LED panel is not a valid solution)

- Is the temperature value you are publishing the same, unaltered numeric value as was released by the SenseHAT? (30 or 29 is not the same as 29.5)
- Are you publishing the latest temperature value? (some solution are queuing temperature value rather than discarding intermediate values)

How to submit

When you're ready to submit, you can upload files for each part of the assignment on the "My submission" tab.