

## Sprint 2

Aim:

To create python script and establish Node-RED

Requirements:

- Python 3.7
- IBM Cloud Account (IoT platform and Node RED App)

Activities:

- Make sure IBM Watson IoT Platform and Node RED App is created in cloud account
- In IBM Watson IoT platform create device and cards
- Develop python script for generating trash bin parameters like weight of the bin and location and GPS
- In Node RED App ,install necessary node
- To retrieve data from IoT Platform use IBMIoT node from node column on left and once double click select the necessary properties
- Place function nodes to get the values ,then type code and connect them to IBMIoT node
- Place debug node and connect them to the function nodes to view the data in debug

Browse

Action

Device Types

Interfaces

Add Device

## Browse Devices

All Devices

Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Q

Search by Device ID

Device Simulator

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added
> <input type="checkbox"/>	1436	Disconnected	preethi	Device	Nov 18, 2022 12:29 AM

Items per page 50 | 1-1 of 1 item

0 Simulations running

IBM

VimelUCh

VimelUCh

Application Details

Node-RED: node-red

IBM Watson IoT Platform

node-red-ubjcz-2022-11-09.au-syd.mybluemix.net/red/#flow/a19a29cee55dd3b

Gmail

YouTube

Maps

New Tab

MIT App Inventor

Node-RED

Inject

Debug

Complete

Catch

Status

Link In

Link Call

Link Out

Comment

Function

Flow 1

IBM IoT

Ultrasonic

Weight

GPS

msg.payload

get /trashbin

data

http

debug

all nodes

all

11/20/2022, 3:47:20 AM node: ID649a-0a0a98

id:28type:preethi1436v4567SensorRedJun :

msg.payload: number

28

11/20/2022, 3:47:21 AM node: ID649a-0a0a98

id:28type:preethi1436v4567SensorRedJun :

msg.payload: number

69

11/20/2022, 3:47:22 AM node: ID649a-0a0a98

id:28type:preethi1436v4567SensorRedJun :

msg.payload: string[10]

"11.391883,76.851592"

50

11/20/2022, 3:47:26 AM node: ID649a-0a0a98

id:28type:preethi1436v4567SensorRedJun :

msg.payload: number

99

11/20/2022, 3:47:26 AM node: ID649a-0a0a98

id:28type:preethi1436v4567SensorRedJun :

msg.payload: number

99

Type here to search

27°C

Haze

8:50

20/11/2022

Node-RED interface showing a flow with an IBM IoT node connected to a function node. The function node is named "Ultrasonic" and contains the following code:

```
1 msg.payload=msg.payload.Ultrasonic;
2 global.set("U",msg.payload);
3 return msg;
```

The debug console shows the following messages:

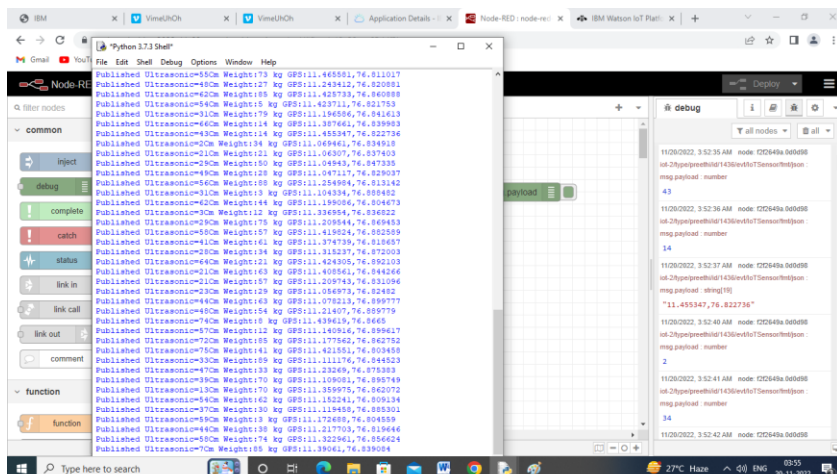
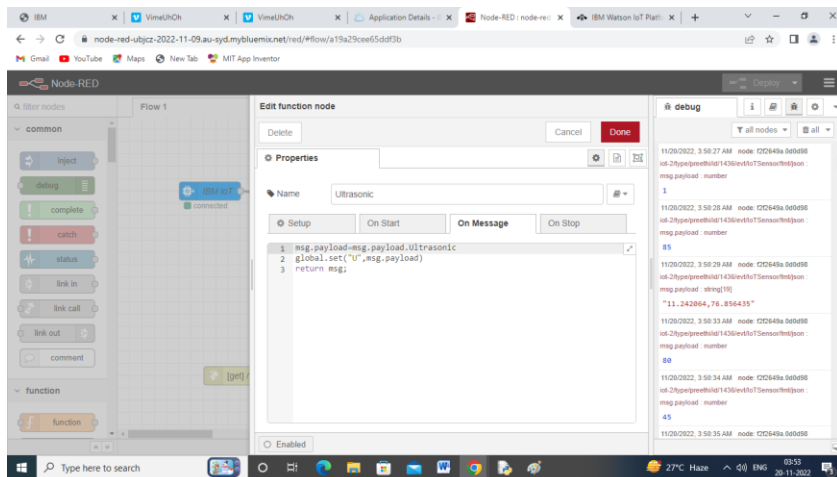
```
11/29/2022, 3:58:27 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
1
11/29/2022, 3:58:28 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
85
11/29/2022, 3:58:29 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: string[10]
"11.242864,76.856435"
11/29/2022, 3:58:33 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
89
11/29/2022, 3:58:34 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
45
11/29/2022, 3:58:35 AM node: ID649a:0d0d98
```

Node-RED interface showing a flow with an IBM IoT node connected to a function node. The function node is named "ibmiot" and contains the following code:

```
1 msg.payload=msg.payload.ibmiot;
2 global.set("U",msg.payload);
3 return msg;
```

The debug console shows the following messages:

```
11/29/2022, 3:48:51 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
5
11/29/2022, 3:48:52 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
90
11/29/2022, 3:48:53 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: string[10]
"11.155333,76.89927"
11/29/2022, 3:48:56 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
25
11/29/2022, 3:48:57 AM node: ID649a:0d0d98
id:28pgipeethid1430ev6t7SensorFeedpon :
msg.payload: number
76
11/29/2022, 3:48:58 AM node: ID649a:0d0d98
```



## Python script

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

```
organization="7lnn7p"
```

```
devicetype="preethi"
```

```
deviceid="1436"
```

```
authMethod="token"
```

```
authToken="09876543211"
```

```
def myCommandCallback(cmd):
```

```
    print("Command received:%s"%cmd.data['command'])
```

```
    status=cmd.data['command']
```

```
    if status == "lighton":
```

```
        print("led in on")
```

```
    else:
```

```
        print("led is off")
```

```
try:
```

```
deviceOptions={ "org":organization,"type":devicetype,"id":deviceid,"  
auth-method":authMethod,"auth-token":authToken }
```

```
    deviceCli=ibmiotf.device.Client(deviceOptions)
```

```
except Exception as e:
```

```
    print("Caught exception connecting device:%s"%str(e))
```

```
    sys.exit()
```

```
deviceCli.connect()
```

```
while True:
```

```
    time.sleep(5)
```

```
    Ultrasonic=random.randint(0,80)
```

```
    Weight=random.randint(0,100)
```

```
    lat=round(random.uniform(11.03,11.50),6)
```

```
    long=round(random.uniform(76.80,76.90),6)
```

```
    GPS=str(lat)+str(',')+str(long)
```

```
    myData={'Ultrasonic':Ultrasonic,'Weight':Weight,'GPS':GPS }
```

```
    def myOnpublishCallback():
```

```
        print("Published Ultrasonic=%sCm"%Ultrasonic,"Weight:%s  
kg"%Weight,"GPS:%s"%GPS)
```

```
success=deviceCli.publishEvent("IoTSensor","json",data=myData,qs=0,on_publish=myOnpublishCallback)
```

```
    if not success:
```

```
        print("Not connected to IoTTF")
```

```
    time.sleep(1)
```

```
    deviceCli.commandCallback=myCommandCallback
```

```
deviceCli.disconnect()
```