|  |
| --- |
| #Provide your IBM Watson Device Credentials organization = "8gyz7t" # replace the ORG ID deviceType = "weather\_monitor" # replace the Device type deviceId = "b827ebd607b5" # replace Device ID authMethod = "token" authToken = "LWVpQPaVQ166HWN48f" # Replace the authtoken |
|  |  |
|  | def myCommandCallback(cmd): # function for Callback if cm.data['command'] == 'motoron': |
|  | print("MOTOR ON IS RECEIVED") |
|  | elif cmd.data['command'] == 'motoroff': print("MOTOR OFF IS RECEIVED") |
|  | if cmd.command == "setInterval": |
|  |  |
|  | else: |
|  | if 'interval' not in cmd.data: |
|  | print("Error - command is missing requiredinformation: 'interval'") |
|  |  |
|  | interval = cmd.data['interval'] |
|  |  |
|  | elif cmd.command == "print": |
|  | if 'message' not in cmd.data: |
|  | print("Error - commandis missing requiredinformation: 'message'") else:output = cmd.data['message'] |
|  | print(output) |
|  |  |
|  |  |
|  | try: |
|  |  |
|  | deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "authmethod": authMethod, |
|  | "auth-token": authToken} deviceCli |
|  | = ibmiotf.device.Client(deviceOptions) # .............................................. |
|  |  |
|  | exceptException as e: |
|  | print("Caught exception connecting device: %s" % str(e)) sys.exit() |
|  |  |
|  | # Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times |
|  | deviceCli.connect() |
|  |  |
|  | while True: |
|  | deviceCli.commandCallback = myCommandCallback |
|  |  |
|  | # Disconnect the device and application from the cloud deviceCli.disconnect() |
|  |  |
|  | SENSOR.PY |
|  |  |
|  | import time import sysimport ibmiotf.application importibmiotf.device |
|  | import random |
|  |  |
|  | # Provide your IBM Watson Device Credentials organization = "8gyz7t" # replace the ORG ID deviceType = "weather\_monitor" # replace the Device type deviceId = "b827ebd607b5" # replace Device ID authMethod = "token" authToken = "LWVpQPaVQ166HWN48f" # Replace the authtoken |
|  |  |
|  |  |
|  |  |
|  | def myCommandCallback(cmd): |
|  |  |
|  | print("Command received: %s" % cmd.data['command']) print(cmd) |
|  |  |
|  | try: |
|  | deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, |
|  | "auth-method": authMethod, "auth-token": authToken} deviceCli = ibmiotf.device.Client(deviceOptions) |
|  | #.............................................. |
|  |  |
|  | exceptException as e: |
|  | print("Caught exception connecting device: %s" % str(e)) sys.exit() |
|  |  |
|  | # Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times |
|  | deviceCli.connect() |
|  |  |
|  | while True: |
|  | temp=random.randint(0,100) pulse=random.randint(0,100) |
|  | soil=random.randint(0,100) |
|  |  |
|  | data = { 'temp' : temp, 'pulse': pulse ,'soil':soil} #print data def |
|  | myOnPublishCallback(): |
|  | print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % pulse,"Soil Moisture = %s %%" % soil,"to IBM Watson") |
|  |  |
|  | success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on\_publish=myOnPublishCallback) if not success: |
|  | print("Not connected to IoTF") time.sleep(1) |
|  |  |
|  | deviceCli.commandCallback = myCommandCallback |
|  |  |
|  | # Disconnect the device and application from the cloud deviceCli.disconnect() |