SOURCE CODE

Team ID :PNT2022TMID34762

```
import time
import sys
 import ibmiotf.application
 import ibmiotf.device
 import random
 #Provide your IBM Watson Device Credentials
organization = "s8ov1q"
 deviceType = "abcd"
 deviceId = "12345"
 authMethod = "token"
authToken = "12345678"
 # Initialize GPIO
def myCommandCallback(cmd):
 print("Command received: %s" % cmd.data['command'])
 status=cmd.data['command']
 if status=="sprinkleron":
  print ("Sprinkler is on")
 elif status == "sprinkleroff":
  print ("Sprinkler is off") elif status == "exhaustfanon": print ("Exhaust Fan ON")
elif status == "exhaustfanoff":
print ("Exhaust Fan OFF")
```

```
 #print(cmd)
 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()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an
event of type "greeting" 10 times
 deviceCli.connect()

while True:
 #Get Sensor Data from DHT11

 temp=random.randint(0,100)
 flame_level=random.randint(0,100)
gas_level = random.randint(0,100)
 data ={ 'Temperature' : temp, 'Flame_Level' : flame_level, 'Gas_Level' : gas_level
}
#print data
def myOnPublishCallback():
     print ("Published Temperature = %s C" % temp, "Flame_Level = %s %%" %
flame_level, "Gas_Level = %s %%" %gas_level ,"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()
import time
import sys
 import ibmiotf.application
 import ibmiotf.device
 import random
 #Provide your IBM Watson Device Credentials
organization = "s8ov1q"
 deviceType = "abcd"
 deviceId = "12345"
 authMethod = "token"
authToken = "12345678"
 # Initialize GPIO
def myCommandCallback(cmd):
 print("Command received: %s" % cmd.data['command'])
 status=cmd.data['command']
 if status=="sprinkleron":
  print ("Sprinkler is on")
 elif status == "sprinkleroff":
  print ("Sprinkler is off") elif status == "exhaustfanon": print ("Exhaust Fan ON")
elif status == "exhaustfanoff":
print ("Exhaust Fan OFF")
```

```
 #print(cmd)
 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()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an
event of type "greeting" 10 times
 deviceCli.connect()

while True:
 #Get Sensor Data from DHT11

 temp=random.randint(0,100)
 flame_level=random.randint(0,100)
gas_level = random.randint(0,100)
 data ={ 'Temperature' : temp, 'Flame_Level' : flame_level, 'Gas_Level' : gas_level
}
#print data
def myOnPublishCallback():
     print ("Published Temperature = %s C" % temp, "Flame_Level = %s %%" %
flame_level, "Gas_Level = %s %%" %gas_level ,"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()
import time
import sys
 import ibmiotf.application
 import ibmiotf.device
 import random
 #Provide your IBM Watson Device Credentials
organization = "s8ov1q"
 deviceType = "abcd"
 deviceId = "12345"
 authMethod = "token"
authToken = "12345678"
 # Initialize GPIO
def myCommandCallback(cmd):
 print("Command received: %s" % cmd.data['command'])
 status=cmd.data['command']
 if status=="sprinkleron":
  print ("Sprinkler is on")
 elif status == "sprinkleroff":
  print ("Sprinkler is off") elif status == "exhaustfanon": print ("Exhaust Fan ON")
elif status == "exhaustfanoff":
print ("Exhaust Fan OFF")
```

```
 #print(cmd)
 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()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an
event of type "greeting" 10 times
 deviceCli.connect()

while True:
 #Get Sensor Data from DHT11

 temp=random.randint(0,100)
 flame_level=random.randint(0,100)
gas_level = random.randint(0,100)
 data ={ 'Temperature' : temp, 'Flame_Level' : flame_level, 'Gas_Level' : gas_level
}
#print data
def myOnPublishCallback():
     print ("Published Temperature = %s C" % temp, "Flame_Level = %s %%" %
flame_level, "Gas_Level = %s %%" %gas_level ,"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()
```