Source Code:

Tank:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "ptrr0s"
deviceType = "Obstacle"
deviceId = "100"
authMethod = "token"
authToken = "12345678"
dis = 0
def myOnPublishCallback():
    print("Published Level = %s %%" % dis, "to IBM Watson")

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status = cmd.data['command']
    if status == "MotorOn":
        print("Motor is running")
    elif status == "MotorOff":
        print("Motor is switched Off")
    else:
        print("please enter a valid command")

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:

      dis =random.randint(0,100)

      data = { 'Distance' : dis}

      success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)

      if dis > 95 :

         print("Motor is switched Off, Tank Full!")

      if not success:

         print("Not connected to IoTF")

      deviceCli.commandCallback = myCommandCallback

      time.sleep(10)


# Disconnect the device and application from the cloud

deviceCli.disconnect()
```

Field:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "ptrr0s"
deviceType = "Field1"
deviceId = "1001"
deviceType1 = "Field2"
deviceId1 = "1002"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

'''elif status == "Servo2On":
        print ("Control valve for Field2 is Open!")
    elif status == "Servo2Off":
        print ("Control valve for Field2 is Closed!")'''

def myCommandCallback1(cmd):
    #print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status == "Servo2On":
        print ("Control valve for Field2 is Open!")
    elif status == "Servo2Off":
        print ("Control valve for Field2 is Closed!")
    #else:
    #    print("please enter a valid command")
```

```
def myCommandCallback(cmd):
   #print("Command received: %s" % cmd.data['command'])
   status=cmd.data['command']
   if status == "Servo1On":
      print ("Control valve for Field1 is Open!")
   elif status == "Servo1Off":
      print ("Control valve for Field1 is Closed!")
   #else:
   #    print("please enter a valid command")


   #print(cmd)


try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-
method": authMethod, "auth-token": authToken}
        deviceOptions1 = {"org": organization, "type": deviceType1, "id": deviceId1, "auth-
method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        deviceCli1 = ibmiotf.device.Client(deviceOptions1)
        #..............................................
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()
deviceCli1.connect()
```

```
def myOnPublishCallback():
   #print("Published Moisture = %s %%" % soil, "Nitrogen = %s %%" % N)#, "RandNo = %s
%%" % No, "to IBM Watson")
   print()
while True:
      soil = random.randint(0,100)
      N = random.randint(0,100)
      P = random.randint(0,100)
      Ka = random.randint(0,100)
      data = { 'Moisture' : soil, 'Nitrogen': N, 'Phosphorus' : P, 'Potassium' : Ka}#,
'randomNumber' : No}
      #print("-----Field 1 Parameters-----")
      success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
      soil = random.randint(0,100)
      N = random.randint(0,100)
      P = random.randint(0,100)
      Ka = random.randint(0,100)
      data = { 'Moisture' : soil, 'Nitrogen': N, 'Phosphorus' : P, 'Potassium' : Ka}
      #print("-----Field 2 Parameters-----")
      success1 = deviceCli1.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
      if not success:
         print("Not connected to IoTF")
      deviceCli.commandCallback = myCommandCallback
      deviceCli1.commandCallback = myCommandCallback1
      time.sleep(10)
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```