```python
# Copyright (c) Microsoft. All rights reserved.
# Licensed under the MIT license. See LICENSE file in the project root for full license information.

import random
import time
from sensor import SHT20

# Using the Python Device SDK for IoT Hub:
#   https://github.com/Azure/azure-iot-sdk-python
# The sample connects to a device-specific MQTT endpoint on your IoT Hub.
from azure.iot.device import IoTHubDeviceClient, Message

# The device connection string to authenticate the device with your IoT hub.
# Using the Azure CLI:
# az iot hub device-identity show-connection-string --hub-name {YourIoTHubName} --device-id MyNodeDevice --output table
CONNECTION_STRING = "HostName=ucliotproject.azure-devices.net;DeviceId=sensorsql;SharedAccessKey=2Y3cIJNZAXsR4nu0SVE+fUy1MuXwtizbe0sdVId1CsE="

#Sensor info

sht = SHT20(1, 0x40)

# Define the JSON message to send to IoT Hub.

MSG_TXT = '{{"temperature": {temperature},"humidity": {humidity}}}'

# Code for connecting to AZURE Clound services

def iothub_client_init():
    # Create an IoT Hub client
    client = IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)
    return client

def iothub_client_telemetry_sample_run():

    try:
        client = iothub_client_init()
        print ( "IoT Hub device sending periodic messages, press Ctrl-C to exit" )
        while True:
            TEMPERATURE = sht.temperature()
            HUMIDITY = sht.humidity()
            temperature = TEMPERATURE.C
            humidity = HUMIDITY.RH
            msg_txt_formatted = MSG_TXT.format(temperature=temperature, humidity=humidity)
            message = Message(msg_txt_formatted)

            # Add a custom application property to the message.
            # Displaying the temperature and humidity data
            message.custom_properties["temperature"] = temperature
            message.custom_properties["humidity"] = humidity

            if temperature > 30:
              message.custom_properties["temperatureAlert"] = "true"
            else:
              message.custom_properties["temperatureAlert"] = "false"

            # Send the message.
            print( "Sending message: {}".format(message) )
            client.send_message(message)
            print ( "Message successfully sent" )
            time.sleep(3)

    except KeyboardInterrupt:
        print ( "IoTHubClient sample stopped" )

if __name__ == '__main__':
    print ( "IoT Hub Quickstart #1 - Simulated device" )
    print ( "Press Ctrl-C to exit" )
    iothub_client_telemetry_sample_run()
```