



mpram Digital Twins HOL

🕒 History

👤 2 contributors



495 lines (272 sloc) | 19.9 KB



# Internet of Things - Azure Digital Twins HOL

1. Azure Digital Twins, theory, deck presented available at **pdf files** folder.

Before Starting this Lab make sure you complete the steps specified in **Azure Digital Twins BHOL.md** File.

## Architecture Diagram

Here

## Content:

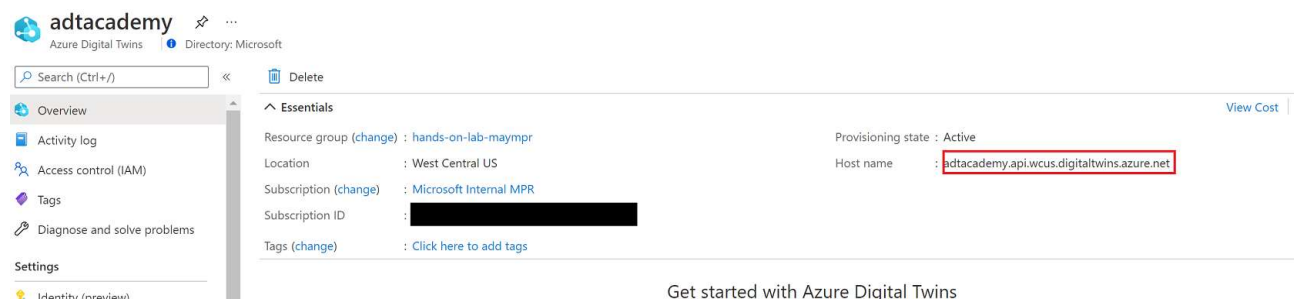
- [Exercise #1: Configure Sample Application](#)
  - [Task 1: Configure sample App](#)
  - [Task 2: Configure Digital Twin Explorer](#)
- [Exercise #2: Process data into Azure Digital Twins](#)
  - [Task #1: Set up sample Function App](#)
  - [Task #2: Process telemetry data](#)

- Exercise #3: Propagate Azure Digital Twins events through the graph
  - Task #1: Set up endpoint
  - Task #2: Set up route
  - Task #3: Connect the function to Event Grid
- Exercise #4: Clean up

## Exercise #1: Configure Sample Application

### Task 1: Configure sample App

1. Open Azure portal and copy the hostname of your Digital Twins created through the steps provided in **Azure Digital Twins BHOL**. Paste the hostname in a notepad.



2. Next, go to the Virtual Machine created during BHOL section, connect using RDP. Go to <https://github.com/Azure-Samples/digital-twins-samples/tree/master/>

Download the code as a zip, unzip the files and look for **AdtSampleApp** Folder. Right click on **AdtE2ESample.sln** and open with in **Visual Studio 2019**.

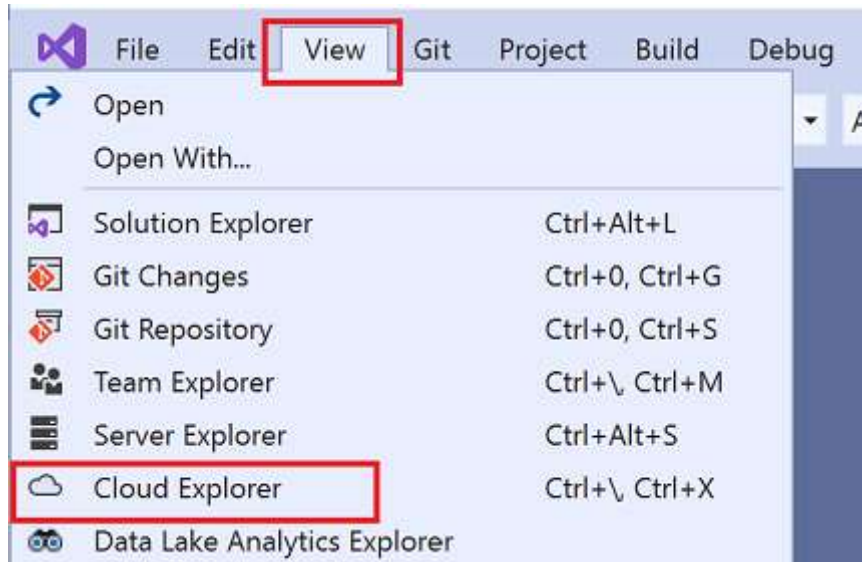
3. Signin in Visual Studio. Once you are in Visual Studio, select the **SampleClientApp** > **appsettings.json** file to open it in the editing window. This will serve as a pre-set JSON file with the necessary configuration variables to run the project.



Note: make sure to add "https://" in front

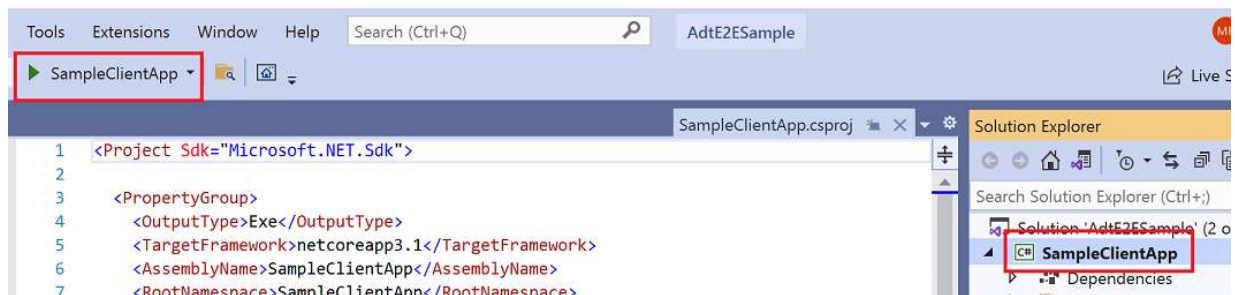
4. Save and close the file

5. In Visual Studio go to the **View Menu** select **Cloud Explorer** as shown below:



In the next window you should see your Subscription or login to your Azure Account.

6. After you are login to Azure, run the **Sample Client App** selecting the button below:



7. A console window will open, carry out authentication, and wait for a command. In this console, run the next command to instantiate the sample Azure Digital Twins solution.

```
C:\Users\iotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\AdtSampleApp\Sam
Authenticating...
Service client created - ready to go

This sample app lets you construct a simple digital twins graph
and issue some commands against the ADT service instance
*** See the command implementation for usage examples of the ADT C# SDK
See the sample documentation for instruction on how to set up additional services
to run an end-to-end demo

Please enter a command or 'help'. Commands are not case sensitive
_
```

8. Copy and paste the following command to

```
SetupBuildingScenario
```

The output of this command is a series of confirmation messages as three digital twins are created and connected in your Azure Digital Twins instance: a floor named floor1, a room named room21, and a temperature sensor named thermostat67. These digital twins represent the entities that would exist in a real-world environment.

They are connected via relationships into the following twin graph. The twin graph represents the environment as a whole, including how the entities interact with and relate to each other.

9. Now your first graph is ready to analyze, to see your graph go to open the digital twin explorer.

## Task 2: Configure Digital Twin Explorer

1. First, download and install Azure CLI using this lin: <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-windows> click on **Current release of the Azure CLI** download and install.

Then open **Windows Powershell** and enter the follow command to login to Azure, the digital twin explorer will use this credentials.

```
az login
```

If you are using multiple subscriptions make sure to set up the subscription properly with the following command

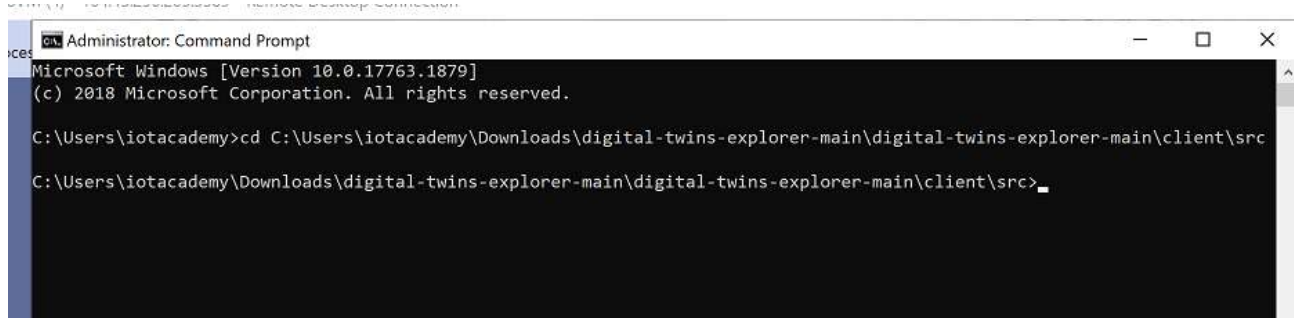
```
az account set --subscription <YOUR SUBSCRIPTION ID HERE>
```

Keep this window open we will explore our twin later here later on.

Second, Download and install **Google Chrome**

Then, Open a cmd window and cd the folder where you download and unzip the digital-twins-explorer-main.zip

```
cd C:\Users\iotacademy\Downloads\digital-twins-explorer-main\digital-twins-explorer-
```

A screenshot of a Windows Command Prompt window titled "Administrator: Command Prompt". The window shows the following text: "Microsoft Windows [Version 10.0.17763.1879] (c) 2018 Microsoft Corporation. All rights reserved. C:\Users\iotacademy>cd C:\Users\iotacademy\Downloads\digital-twins-explorer-main\digital-twins-explorer-main\client\src C:\Users\iotacademy\Downloads\digital-twins-explorer-main\digital-twins-explorer-main\client\src>". The command prompt is open in a black window with white text.

Once in the **src** folder run

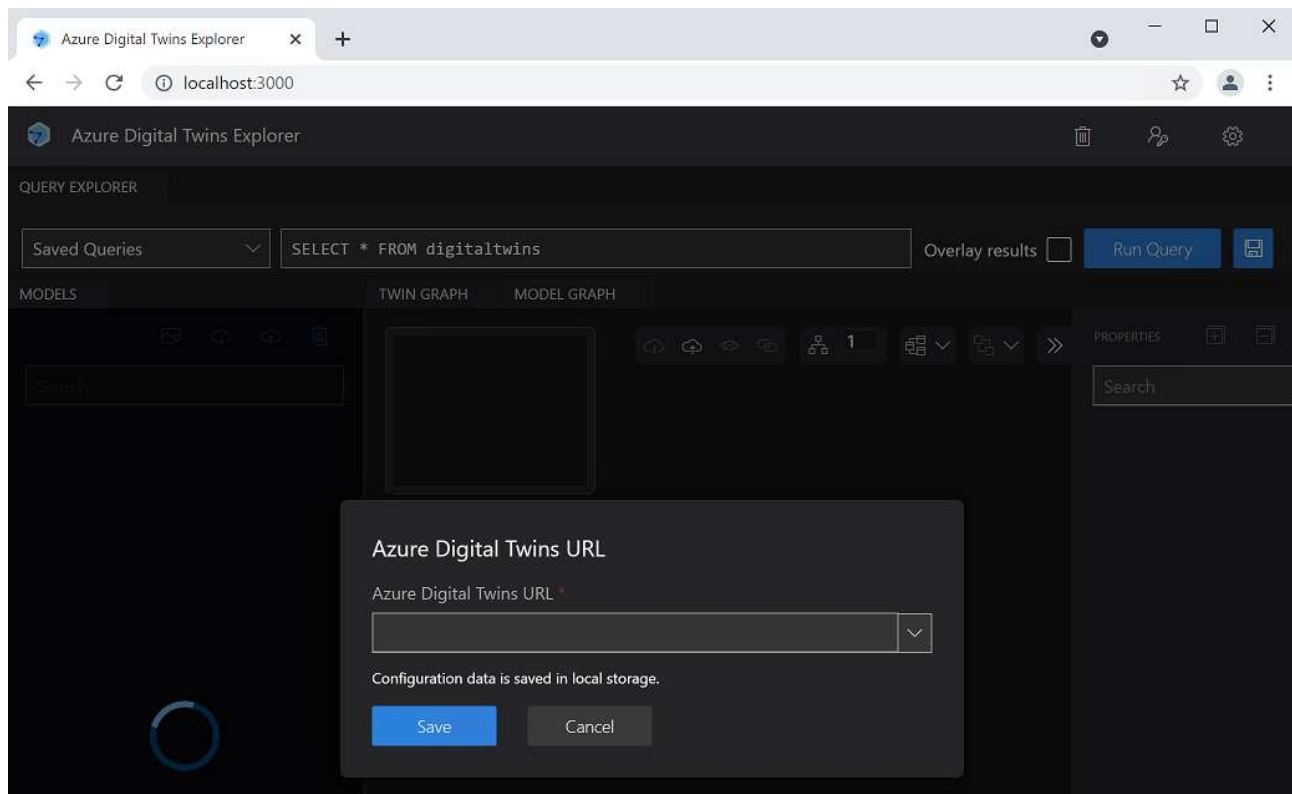
```
npm run start
```

**Note:** In Before HOL we asked you to install npm, if you have an error here, run **npm install** first, assuming you download and install node.js first. Then go back and run **npm run start**

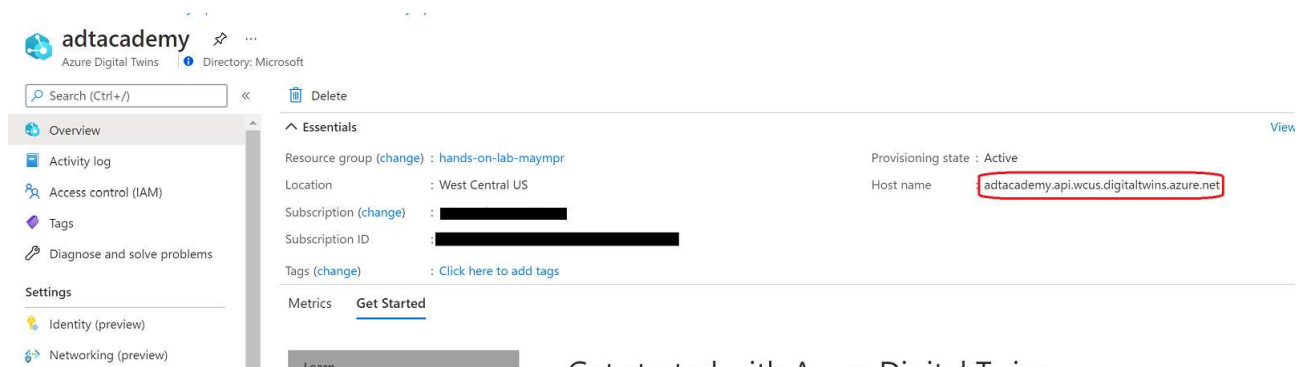
2. Now open Google Chrome and copy the following url:

<http://localhost:3000/>

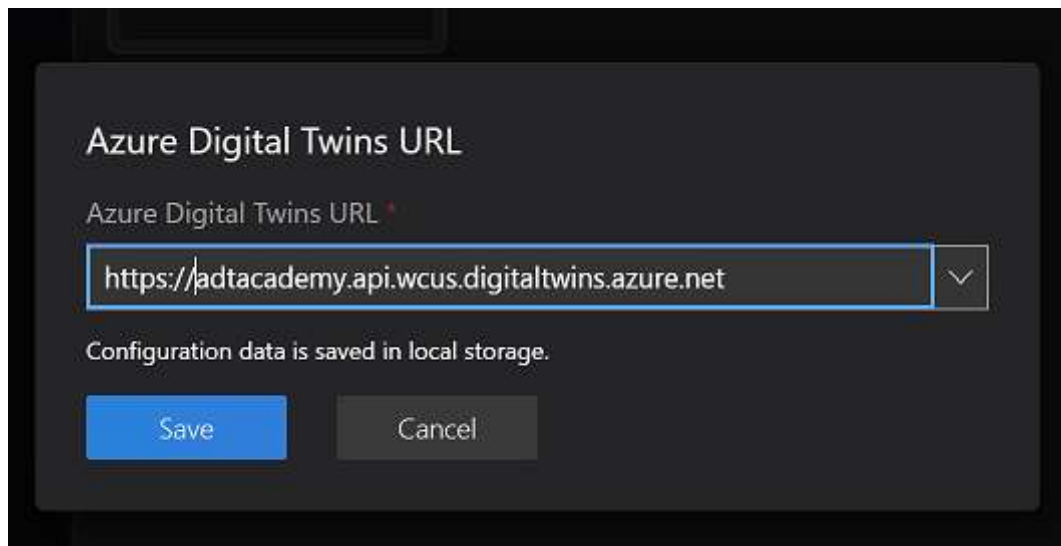
You should be able to see the app as shown below:



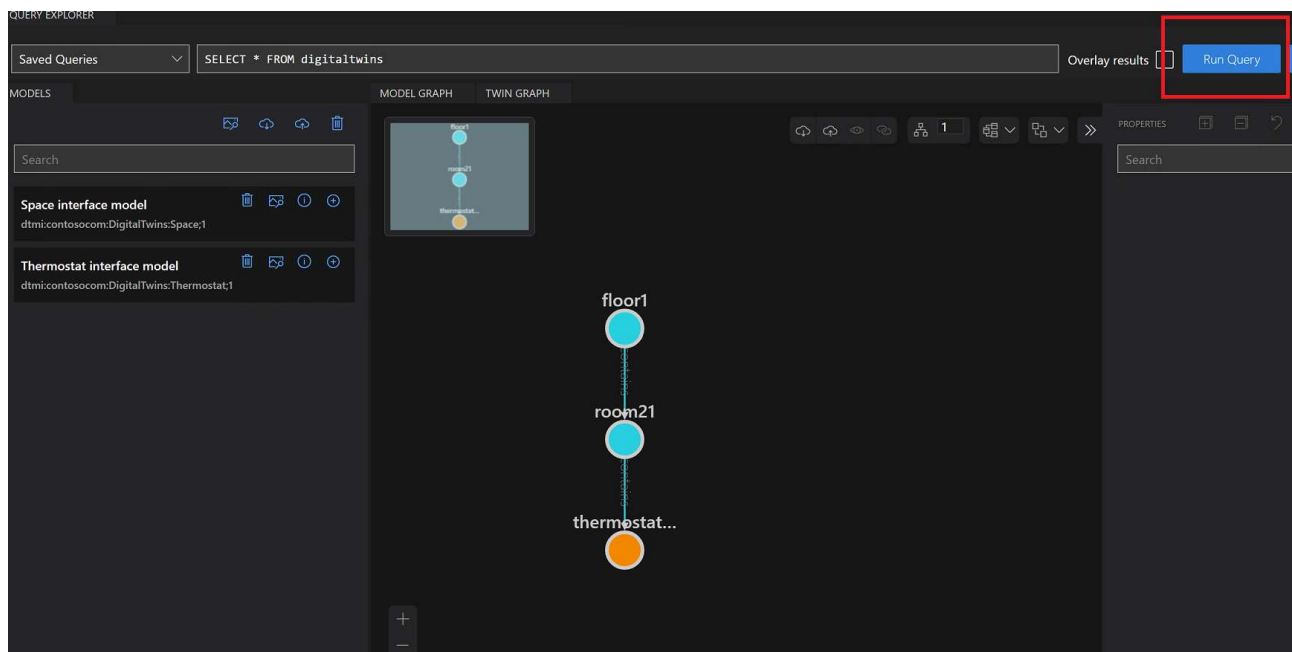
3. Copy the adt hostname from Azure Portal, add **https://** and paste in the Digital Twin Explorer app.



Your App should look like below and then click **Save**



3. Once login, run the default query clicking in **Run query** and you should see the graph just created:



Take a few minutes to explore the tool.

4. Go back to powershell and run and run the following commnad to see the definition of your Digital Twin.

```
az dt model list -n <ADT_instance_name> --definition
```



```

PS C:\Users\iotacademy> az dt model list -n adtacademy.api.wcus.digitaltwins.azure.net
The command requires the extension azure-iot. Do you want to install it now? The command will continue to run after the
extension is installed. (Y/n): Y
Run 'az config set extension.use_dynamic_install=yes_without_prompt' to allow installing extensions without prompt.
[[{"Installing": true,
{
  "decommissioned": false,
  "description": {},
  "displayName": {
    "en": "Thermostat interface model"
  },
  "id": "dtmi:contosocom:DigitalTwins:Thermostat;1",
  "model": null,
  "uploadTime": "2021-04-23T19:38:43.035214+00:00"
},
{
  "decommissioned": false,
  "description": {},
  "displayName": {
    "en": "Space interface model"
  },
  "id": "dtmi:contosocom:DigitalTwins:Space;1",
  "model": null,
  "uploadTime": "2021-04-23T19:38:43.035270+00:00"
}
}]
PS C:\Users\iotacademy>

```

5. You can also use CLI to update, delete or query your twin. Run the following command to query your twin.

```
az dt twin query -n <ADT_instance_name> -q "SELECT * FROM DIGITALTWINS"
```

```

PS C:\Users\iotacademy> az dt twin query -n adtacademy.api.wcus.digitaltwins.azure.net -q "SELECT * FROM DIGITALTWINS"
{
  "result": [
    {
      "$dtId": "floor1",
      "$etag": "W/\"5a49f262-d062-45e3-a344-49ba45021ee6\"",
      "$metadata": {
        "$model": "dtmi:contosocom:DigitalTwins:Space;1",
        "ComfortIndex": {
          "lastUpdateTime": "2021-04-29T14:13:01.5967537Z"
        },
        "DisplayName": {
          "lastUpdateTime": "2021-04-29T14:13:01.5967537Z"
        },
        "Location": {
          "lastUpdateTime": "2021-04-29T14:13:01.5967537Z"
        },
        "Temperature": {
          "lastUpdateTime": "2021-04-29T14:13:01.5967537Z"
        }
      },
      "ComfortIndex": 0,
      "DisplayName": "Floor 1",
      "Location": "Puget Sound",
      "Temperature": 0
    },
    {
      "$dtId": "room21",
      "$etag": "W/\"d4315e36-37d2-48d0-bf9f-b5e9ad1e5ef4\"",
      "$metadata": {
        "$model": "dtmi:contosocom:DigitalTwins:Space;1",
        "ComfortIndex": {
          "lastUpdateTime": "2021-04-29T14:13:01.8280193Z"
        },
        "DisplayName": {
          "lastUpdateTime": "2021-04-29T14:13:01.8280193Z"
        },
        "Location": {
          "lastUpdateTime": "2021-04-29T14:13:01.8280193Z"
        }
      },
      "ComfortIndex": 0,
      "DisplayName": "Room 21",
      "Location": "Puget Sound",
      "Temperature": 0
    }
  ]
}

```

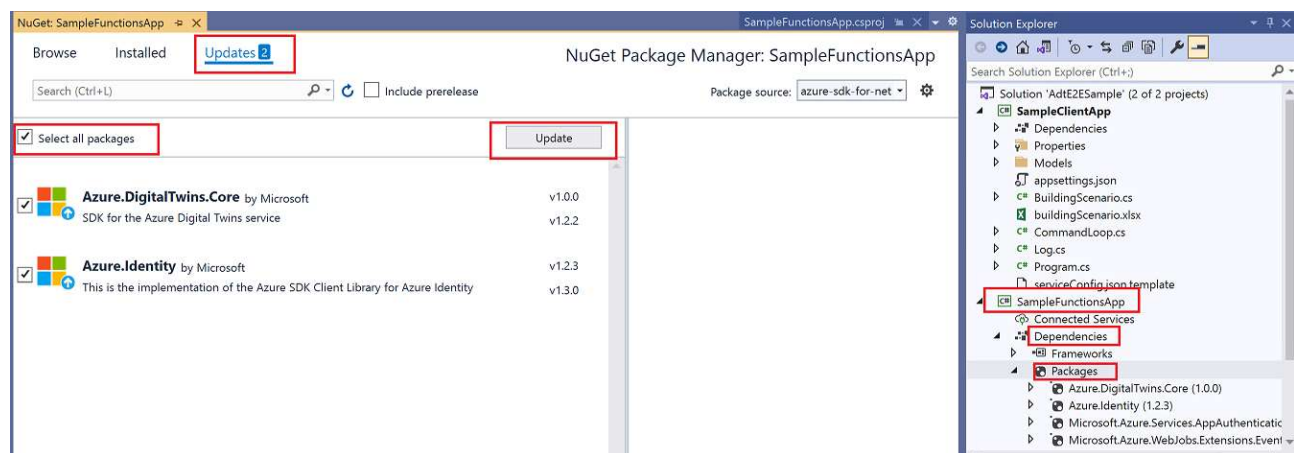
## Exercise #2: Process data into Azure Digital Twins



## Task #1: Set up sample Function App

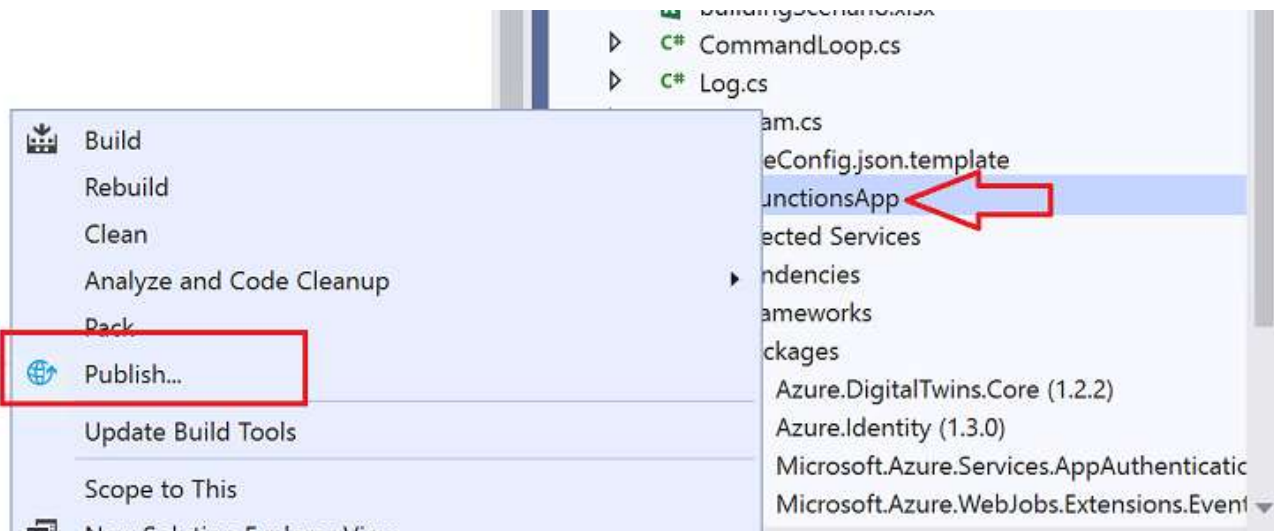
The next step is setting up an Azure Functions app that will be used throughout this tutorial to process data. The function app, `SampleFunctionsApp`, contains two functions: `ProcessHubToDTEvents`: processes incoming IoT Hub data and updates Azure Digital Twins accordingly `ProcessDTRoutedData`: processes data from digital twins, and updates the parent twins in Azure Digital Twins accordingly

1. In Visual Studio, stop the app if still running then go to the Solution Explorer pane, expand **SampleFunctionsApp** > **Dependencies** then Right-select **Packages** and choose **Manage NuGet Packages...**
2. Select the tab **Updates** then click on **Select all packages** and **Update**



A new window will pop up asking for **Preview Changes** click **Ok**. Then click in **I Accept** to the license agreements.

3. Go back to the **SampleFunctionsApp** and right click and select **Publish**



4. On the Publish page that opens, leave the default target selection of **Azure**. Then select **Next**.

For a specific target, choose **Azure Function App (Windows)** and then select **Next**.

In the Publish window select your subscription and click in + to create a new Function App

A screenshot of the 'Publish' window in Visual Studio. The window title is 'Publish'. Below the title, it says 'Select existing or create a new Azure Function'. In the top right corner, there's a dropdown menu showing 'Microsoft' and a redacted email address. The 'Target' section has a dropdown menu with 'Azure' selected. The 'Specific target' section has a dropdown menu with 'Functions instance' selected. Below this, there's a 'View' dropdown menu with 'Resource group' selected. A 'Search' input field is present. Below the search field, there's a list of 'Function Apps' with a red '+' button and a refresh icon to the right. At the bottom, there's a checkbox labeled 'Run from package file (recommended)' which is checked. At the very bottom, there are four buttons: 'Back', 'Next', 'Finish', and 'Cancel'.

The New window will ask you to fill the information for the new Function App

- **Name:** Leave default value.

- **Subscription Name:** The subscription you are using for this training.
- **Resource Group:** The resource group you are using for this training.
- **Plan Type: Consumption**
- **Location:** The location you are using for this training
- **Azure Storage:** click on **New** and create a new storage account for this Function App.

Then click **Create**.

After a few minutes the new instance should appear available, click on **Finish**

In the next window select **Publish**

After a few minutes in your Output window you should see the following message:

===== Publish: 1 succeeded, 0 failed, 0 skipped =====

5. The first setting gives the function app the Azure Digital Twins Data Owner role in the

Azure Digital Twins instance. This role is required for any user or function that wants to perform many data plane activities on the instance.

Go back to Powershell and run the following command:

```
az functionapp identity show -g <YOUR RESOURCE GROUP NAME HERE> -n <YOUR FUNCTION AP
```

Now copy the principal id of your Function App

```
PS C:\Users\iotacademy> az functionapp identity show -g hands-on-lab-maympr -n SampleFunctionsApp20210429170525
{
  "principalId": "06a000be-e67c-40cd-a707-711034acee4c",
  "tenantId": "72de4ee5c-8be8-4a3b-b5bf-4399fdb979c6",
  "type": "SystemAssigned",
  "userAssignedIdentities": null
}
PS C:\Users\iotacademy>
```

6. Use the principalId value in the following command to assign the function app's identity to the Azure Digital Twins Data Owner role for your Azure Digital Twins instance.

If you have access to multiple resource groups in this subscription, is better if you run this command first.

```
az config set defaults.group=<YOUR RESOURCE GROUP HERE>
```

Then run the following command:

```
az dt role-assignment create --dt-name <YOUR ADT NAME HERE> --assignee "<YOUR PRINCIPAL ID>" --role "Azure Digital Twins Data Owner"
```

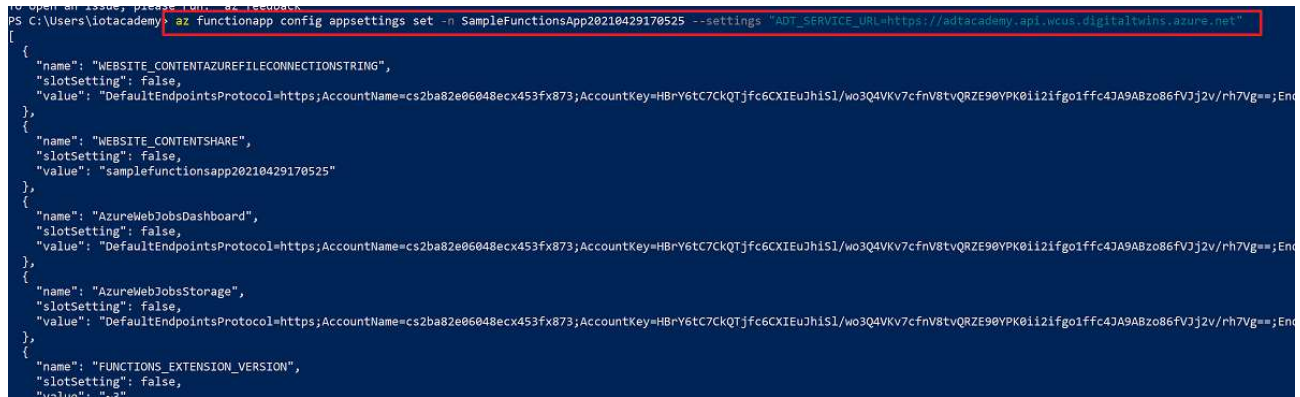
The result of this command is outputted information about the role assignment you've created. The function app now has permissions to access data in your Azure Digital Twins instance.

```
PS C:\Users\iotacademy> az dt role-assignment create --dt-name adtacademy --assignee "06a000be-e67c-40cd-a707-711034acee4c" --role "Azure Digital Twins Data Owner"
{
  "canDelegate": null,
  "condition": null,
  "conditionVersion": null,
  "description": null,
  "id": "/subscriptions/72de4ee5c-8be8-4a3b-b5bf-4399fdb979c6/resourceGroups/hands-on-lab-maympr/providers/Microsoft.DigitalTwins/digitalTwinsInstances/adtacademy/providers/Microsoft.Authorization/roleAssignments/bcd981a7-7f74-457b-83e1-cceb9e632ffe",
  "name": "7de4ee5c-8be8-4a3b-b5bf-4399fdb979c6",
  "principalId": "06a000be-e67c-40cd-a707-711034acee4c",
  "principalType": "ServicePrincipal",
  "resourceGroup": "hands-on-lab-maympr",
  "roleDefinitionId": "/subscriptions/72de4ee5c-8be8-4a3b-b5bf-4399fdb979c6/providers/Microsoft.Authorization/roleDefinitions/bcd981a7-7f74-457b-83e1-cceb9e632ffe",
  "scope": "/subscriptions/72de4ee5c-8be8-4a3b-b5bf-4399fdb979c6/resourceGroups/hands-on-lab-maympr/providers/Microsoft.DigitalTwins/digitalTwinsInstances/adtacademy",
  "type": "Microsoft.Authorization/roleAssignments"
}
PS C:\Users\iotacademy>
```

7. This second setting creates an environment variable for the function with the URL of your Azure Digital Twins instance. The function code will use this to refer to your instance. Run the following command:

```
az functionapp config appsettings set -n <YOUR FUNCTION APP NAME HERE> --settings "A
```

The output should look like the below image:



```
PS C:\Users\iotacademy> az functionapp config appsettings set -n SampleFunctionsApp20210429170525 --settings "ADT_SERVICE_URL=https://adtacademy.api.wcus.digitaltwins.azure.net"
[
  {
    "name": "WEBSITE_CONTENTAZUREFILECONNECTIONSTRING",
    "slotSetting": false,
    "value": "DefaultEndpointsProtocol=https;AccountName=cs2ba82e06048ecx453fx873;AccountKey=HBry6tC7CkQTjfc6CXIEU3hISl/wo3Q4VKv7cfnV8tvQRZE90YPK0i12ifgo1ffc4JA9ABzo86fVj2v/rh7Vg==;End
  },
  {
    "name": "WEBSITE_CONTENTSHARE",
    "slotSetting": false,
    "value": "samplefunctionsapp20210429170525"
  },
  {
    "name": "AzureWebJobsDashboard",
    "slotSetting": false,
    "value": "DefaultEndpointsProtocol=https;AccountName=cs2ba82e06048ecx453fx873;AccountKey=HBry6tC7CkQTjfc6CXIEU3hISl/wo3Q4VKv7cfnV8tvQRZE90YPK0i12ifgo1ffc4JA9ABzo86fVj2v/rh7Vg==;End
  },
  {
    "name": "AzureWebJobsStorage",
    "slotSetting": false,
    "value": "DefaultEndpointsProtocol=https;AccountName=cs2ba82e06048ecx453fx873;AccountKey=HBry6tC7CkQTjfc6CXIEU3hISl/wo3Q4VKv7cfnV8tvQRZE90YPK0i12ifgo1ffc4JA9ABzo86fVj2v/rh7Vg==;End
  },
  {
    "name": "FUNCTIONS_EXTENSION_VERSION",
    "slotSetting": false,
    "value": "~3"
```

## Task #2: Process telemetry data

Here are the actions you will complete to set up this device connection:

- Create an IoT hub that will manage the simulated device
- Connect the IoT hub to the appropriate Azure function by setting up an event subscription
- Register the simulated device in IoT hub
- Run the simulated device and generate telemetry
- Query Azure Digital Twins to see the live results


1. Create an IoT Hub replace the suffix and assing your resource group.

```
az iot hub create --name hubadtlabSUFFIX -g <your-resource-group> --sku S1
```


The output of this command is information about the IoT hub that was created.


Save the name that you gave to your IoT hub. You will use it later.


2. Connect the IoT hub to the Azure function. Once the IoT Hub is created go to the Azure Portal and select the IoT Hub, then click on **Events** then **Events Subscription**


 **hubadtlab | Events** ...


IoT Hub Directory: Microsoft


 Overview


 Activity log


 Access control (IAM)

 Tags

 Diagnose and solve problems

 **Events**

 **Event Subscription**

 Refresh

Get Started

**Event Subscriptions**

This will bring up the Create Event Subscription page.



# Create Event Subscription

Event Grid

Basic

Filters

Additional Features

Delivery Properties

Event Subscriptions listen for events emitted by the topic resource and send them to the endpoint resource. [Learn more](#)

## EVENT SUBSCRIPTION DETAILS

Name \*

adt-event-subscription



Event Schema

Event Grid Schema



## TOPIC DETAILS

Pick a topic resource for which events should be pushed to your destination. [Learn more](#)

Topic Type



IoT Hub

Source Resource

hubadtlab

System Topic Name \*



adt-system-topic



## EVENT TYPES

Pick which event types get pushed to your destination. [Learn more](#)

Filter to Event Types

Device Telemetry



## ENDPOINT DETAILS

Pick an event handler to receive your events. [Learn more](#)

Endpoint Type \*



Azure Function [\(change\)](#)

Endpoint \*

[Select an endpoint](#)

Create

Fill in the fields as follows (fields filled by default are not mentioned):

## EVENT SUBSCRIPTION DETAILS

- **Name:** Give a name to your event subscription.

## TOPIC DETAILS

- **System Topic Name:** Give a name to use for the system topic.



## EVENT TYPES

- Filter to Event Types: Select **Device Telemetry** from the menu options.

## ENDPOINT DETAILS

- Endpoint Type: **Azure Function**

## ENDPOINT DETAILS

- Endpoint: Hit the **Select an endpoint** link. This will open a Select Azure Function window:

### Select Azure Function

Event Grid

Subscription

Resource group

Function app \*

Slot \* ⓘ

Function \* ⓘ

Confirm Selection

Fill in your Subscription, Resource group, Function app and Function (**ProcessHubToDTEvents**). Some of these may auto-populate after selecting the subscription.

Hit **Confirm Selection**. then click on **Create**

3. Go back to the IoT Hub and create a device for your simulator. Select **Explorer**, then **IoT Devices** then **New**

The form to create device will pop up fill the information:.



# Create a device

Directory: Microsoft



Find Certified for Azure IoT devices in the Device Catalog



Device ID \* ⓘ

The ID of the new device

Authentication type ⓘ

Symmetric key

X.509 Self-Signed

X.509 CA Signed

Primary key ⓘ

Enter your primary key

Secondary key ⓘ

Enter your secondary key

Auto-generate keys ⓘ



Connect this device to an IoT hub ⓘ

Enable

Disable

Parent device ⓘ

**No parent device**

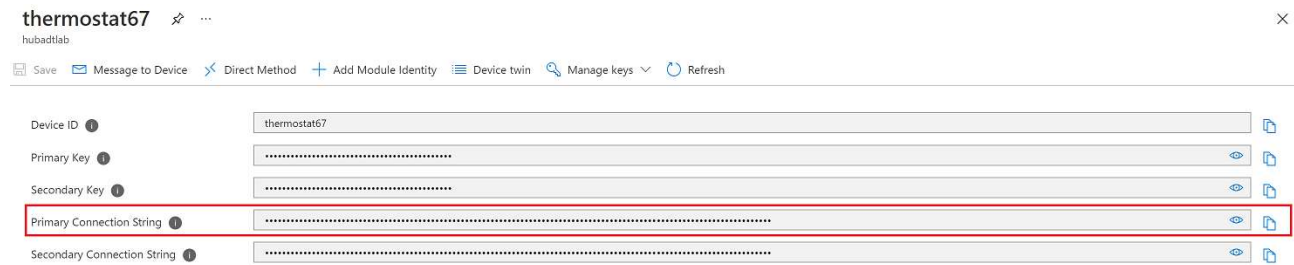
[Set a parent device](#)

Save

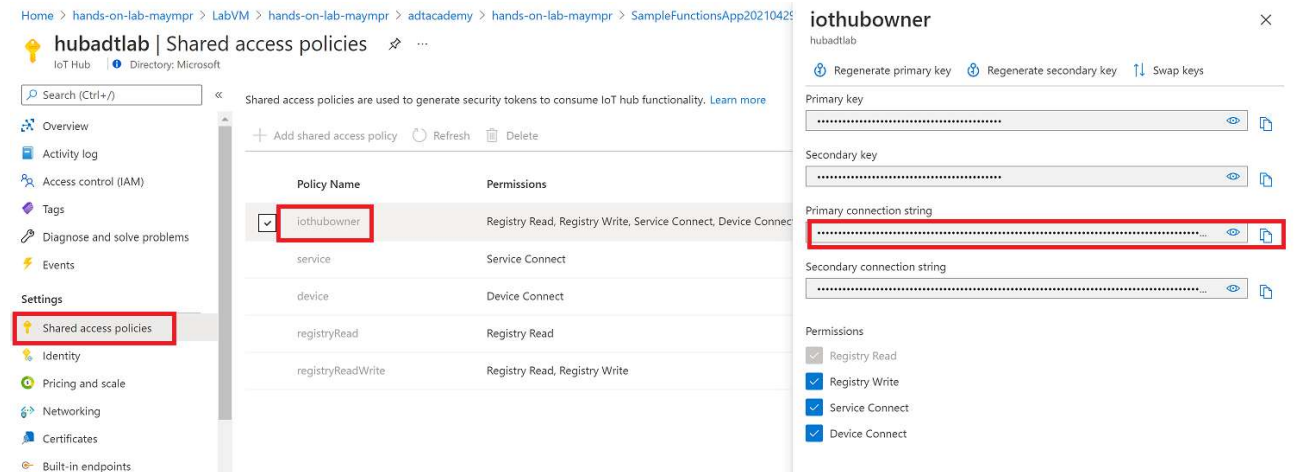
- **Device ID:** thermostat67
- **Authentication type:** Symetric Key
- **Auto-generate keys:** Make sure it is checked
- **Connect this device to an IoT Hub:** Enable

Then click **Save**

Open again the device and copy in a notepad the connection string:

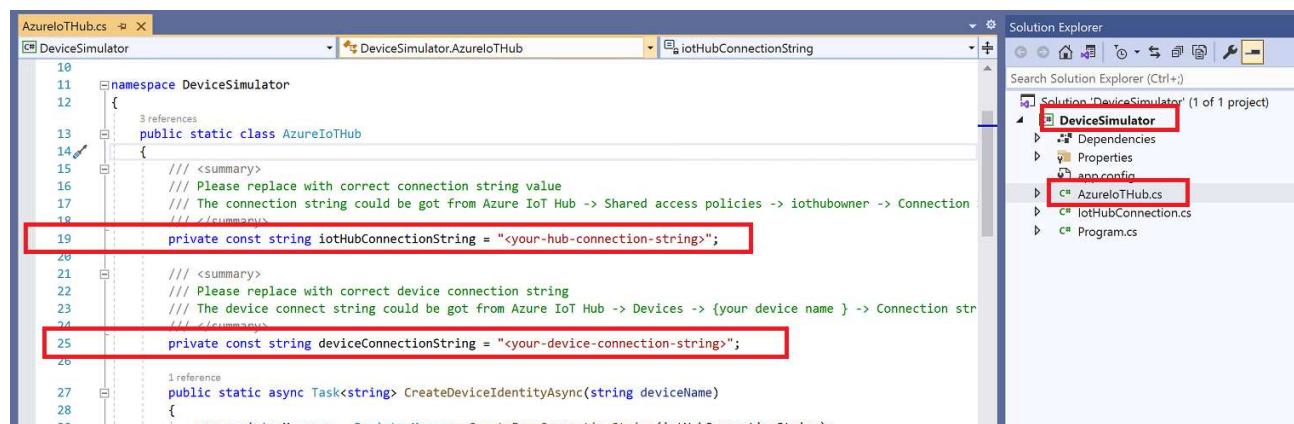


We will also need the IoT Hub Connection String. Go to your IoT Hub, select **Shared Access Policies** in Settings, then **iothubowner** in **Policy Name** copy the **Primary Connection String** also in the notepad.



- In a new Visual Studio window, open **DeviceSimulator.sln**. (from the downloaded solution folder) in this case will be a path similar to this one:  
C:\Users\iotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\DeviceSimulator. Right click on the **DeviceSimulator.sln**. Open with **Visual Studio 2019**

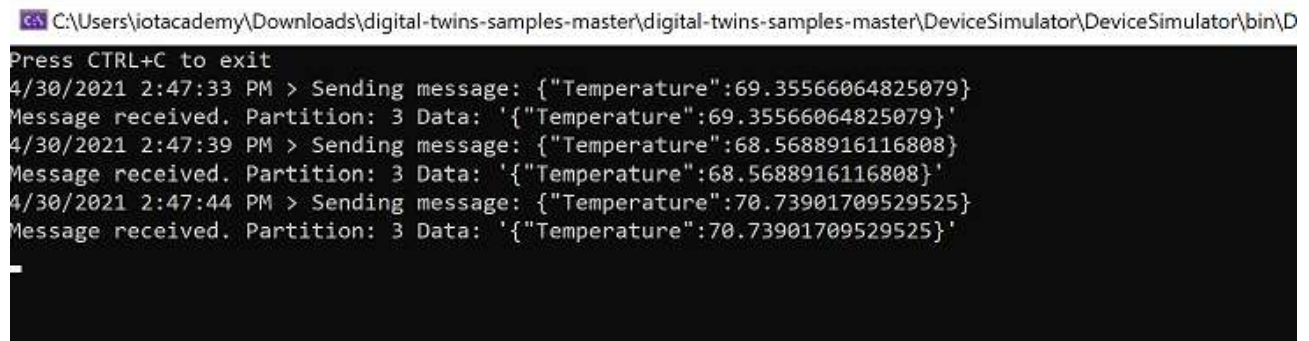
From the Solution Explorer pane in this new Visual Studio window, select **DeviceSimulator/AzureIoTHub.cs** to open it in the editing window. Change the following connection string values to the values you gathered above:



Save the file, Ctrl+S

Now, to see the results of the data simulation that you've set up, run the DeviceSimulator project with the green play button in the toolbar **Start**

A console window will open and display simulated temperature telemetry messages. These are being sent to IoT Hub, where they are then picked up and processed by the Azure function.

A screenshot of a console window with a black background and white text. The window title bar shows the file path: C:\Users\jotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\DeviceSimulator\DeviceSimulator\bin\Debug. The console output shows a series of messages: 'Press CTRL+C to exit', followed by three pairs of 'Sending message' and 'Message received' logs. Each log contains a JSON object with a 'Temperature' key and a numerical value. The values are 69.35566064825079, 68.5688916116808, and 70.73901709529525 respectively.

```
C:\Users\jotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\DeviceSimulator\DeviceSimulator\bin\Debug
Press CTRL+C to exit
4/30/2021 2:47:33 PM > Sending message: {"Temperature":69.35566064825079}
Message received. Partition: 3 Data: '{"Temperature":69.35566064825079}'
4/30/2021 2:47:39 PM > Sending message: {"Temperature":68.5688916116808}
Message received. Partition: 3 Data: '{"Temperature":68.5688916116808}'
4/30/2021 2:47:44 PM > Sending message: {"Temperature":70.73901709529525}
Message received. Partition: 3 Data: '{"Temperature":70.73901709529525}'
```

You don't need to do anything else in this console, but leave it running while you complete the next steps.

5. To see the data from the Azure Digital Twins side, go to your Visual Studio window where the **AdtE2ESample** project is open and run the project.

In the project console window that opens, run the following command to get the temperatures being reported by the digital twin thermostat67:

```
ObserveProperties thermostat67 Temperature
```

You should see the data flowing through the digital twin

```
C:\Users\iotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\AdtSampleApp\Samp
Authenticating...
Service client created - ready to go

This sample app lets you construct a simple digital twins graph
and issue some commands against the ADT service instance
*** See the command implementation for usage examples of the ADT C# SDK
See the sample documentation for instruction on how to set up additional services
to run an end-to-end demo

Please enter a command or 'help'. Commands are not case sensitive
ObserveProperties thermostat67 Temperature
Starting observation...
Press any key to end observation
$dtId: thermostat67, Temperature: 68.92527696021146
$dtId: thermostat67, Temperature: 68.92527696021146
$dtId: thermostat67, Temperature: 68.92527696021146
$dtId: thermostat67, Temperature: 68.75506543822357
$dtId: thermostat67, Temperature: 68.75506543822357
$dtId: thermostat67, Temperature: 68.75506543822357
$dtId: thermostat67, Temperature: 67.51837154315709
$dtId: thermostat67, Temperature: 67.51837154315709
$dtId: thermostat67, Temperature: 67.51837154315709
$dtId: thermostat67, Temperature: 70.71263370835811
$dtId: thermostat67, Temperature: 70.71263370835811
$dtId: thermostat67, Temperature: 70.71263370835811
```

Once you've verified this is working successfully, you can stop running both projects. Keep the Visual Studio windows open, as you'll continue using them in the rest of the tutorial.

## Exercise #3: Propagate Azure Digital Twins events through the graph

Here are the actions you will complete to set up this data flow:

- Create an Event Grid endpoint in Azure Digital Twins that connects the instance to Event Grid.
- Set up a route within Azure Digital Twins to send twin property change events to the endpoint.
- Deploy an Azure Functions app that listens (through Event Grid) on the endpoint, and updates other twins accordingly.
- Run the simulated device and query Azure Digital Twins to see the live results



## Task #1: Set up endpoint

In this section, you create an event grid topic, and then create an endpoint within Azure Digital Twins that points (sends events) to that topic.

1. Go back to the powershell window and run the following command replacing **SUFFIX** and your **Resource Group**

```
az eventgrid topic create -g <YOUR RESOURCE GROUP HERE> --name eventgridtopicadtsUFF
```

```
PS C:\Users\iotacademy> az eventgrid topic create -g hands-on-lab-maympr --name eventgridtopicadtsUFFIX -l eastus
{
  "endpoint": "https://eventgridtopicadtsuffix.eastus-1.eventgrid.azure.net/api/events",
  "extendedLocation": null,
  "id": "/subscriptions/10a3488e-615c-4839-86eb-956854335f6b/resourceGroups/hands-on-lab-maympr/providers/Microsoft.EventGrid/topics/eventgridtopicadtsUFFIX",
  "identity": {
    "principalId": null,
    "tenantId": null,
    "type": "None",
    "userAssignedIdentities": null
  },
  "inboundIpRules": null,
  "inputSchema": "EventGridSchema",
  "inputSchemaMapping": null,
  "kind": "Azure",
  "location": "eastus",
  "metricResourceId": "10a3488e-615c-4839-86eb-956854335f6b",
  "name": "eventgridtopicadtsUFFIX",
  "privateEndpointConnections": null,
  "provisioningState": "Succeeded",
  "publicNetworkAccess": "Enabled",
  "resourceGroup": "hands-on-lab-maympr",
  "sku": {
    "name": "Basic"
  },
  "systemData": null,
  "tags": null,
  "type": "Microsoft.EventGrid/topics"
}
```

2. Next, create an Event Grid endpoint in Azure Digital Twins, which will connect your instance to your event grid topic. Use the command below, filling in the placeholder fields as necessary:

```
az dt endpoint create eventgrid --dt-name <your-Azure-Digital-Twins-instance> --even
```

```

PS C:\Users\iotacademy> az dt endpoint create eventgrid --dt-name adtacademy --eventgrid-resource-group hands-on-lab-maym
mpr --eventgrid-topic eventgridtopicadtsUFFIX --endpoint-name endpointadtlab
{
  "id": "/subscriptions/12345678901234567890/resourceGroups/hands-on-lab-maymmp/providers/Microsoft.DigitalTwins/digitalTwinsInstances/adtacademy/endpoints/endpointadtlab",
  "name": "endpointadtlab",
  "properties": {
    "accessKey1": "****",
    "accessKey2": "****",
    "authenticationType": "KeyBased",
    "createdTime": "2021-04-30T15:39:57.510700+00:00",
    "deadLetterSecret": null,
    "deadLetterUri": null,
    "endpointType": "EventGrid",
    "provisioningState": "Provisioning",
    "topicEndpoint": "https://###"
  },
  "resourceGroup": "hands-on-lab-maymmp",
  "type": "Microsoft.DigitalTwins/digitalTwinsInstances/endpoints"
}
PS C:\Users\iotacademy>

```

The output from this command is information about the endpoint you've created.

Look for the provisioningState field in the output, and check that the value is "Succeeded". It may also say "Provisioning", meaning that the endpoint is still being created. In this case, wait a few seconds and run the command again to check that it has completed successfully before continuing next steps to create routes using this endpoint.

```
az dt endpoint show --dt-name <your-Azure-Digital-Twins-instance> --endpoint-name <y
```

Save the names that you gave to your event grid topic and your Event Grid endpoint in Azure Digital Twins. You will use them later.

## Task #2: Set up route

Next, create an Azure Digital Twins route that sends events to the Event Grid endpoint you just created.

1. Run the following command to create a route, be careful to replace all the values based on your previous steps.

```
az dt route create --dt-name <your-Azure-Digital-Twins-instance> --endpoint-name <y
```

The output from this command is some information about the route you've created.



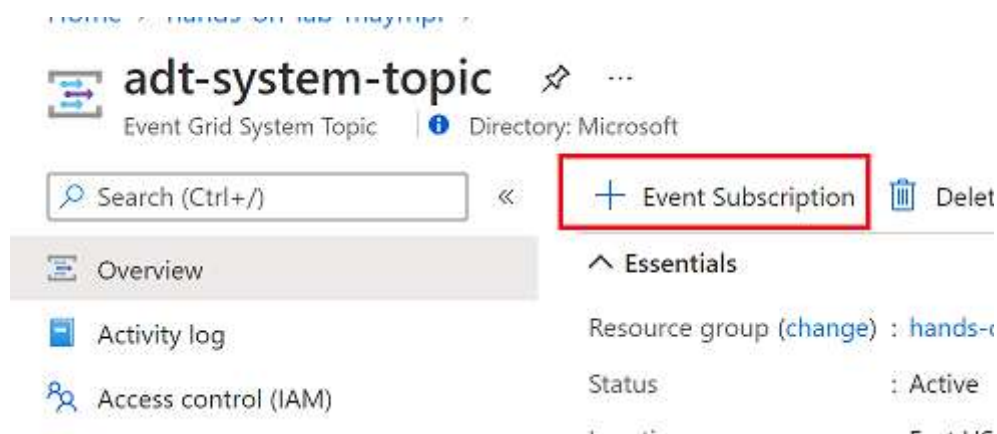
```
PS C:\Users\iotacademy> az dt route create --dt-name adtacademy --endpoint-name endpointadtlab --route-name adtroute
{
  "endpointName": "endpointadtlab",
  "filter": "true",
  "id": "adtroute"
}
PS C:\Users\iotacademy>
```

### Task #3: Connect the function to Event Grid

Next, subscribe the ProcessDTRoutedData Azure function to the event grid topic you created earlier, so that telemetry data can flow from the thermostat67 twin through the event grid topic to the function, which goes back into Azure Digital Twins and updates the room21 twin accordingly.

To do this, you'll create an Event Grid subscription that sends data from the event grid topic that you created earlier to your ProcessDTRoutedData Azure function.

1. Go to Azure Portal, navigate to the Event grid Topic created before, then click **+ Event Subscription**



A new form will open fill the values as shown below:



# Create Event Subscription

Event Grid

Basic

Filters

Additional Features

Delivery Properties

Event Subscriptions listen for events emitted by the topic resource and send them to the endpoint resource. [Learn more](#)

## EVENT SUBSCRIPTION DETAILS

Name \*

routeddataevents



Event Schema

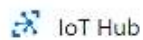
Event Grid Schema



## TOPIC DETAILS

Pick a topic resource for which events should be pushed to your destination. [Learn more](#)

Topic Type



IoT Hub

Source Resource

hubadtlab

Topic Name



adt-system-topic

## EVENT TYPES

Pick which event types get pushed to your destination. [Learn more](#)

Filter to Event Types

4 selected



## ENDPOINT DETAILS

Pick an event handler to receive your events. [Learn more](#)

Endpoint Type \*



Azure Function [\(change\)](#)

Endpoint \*

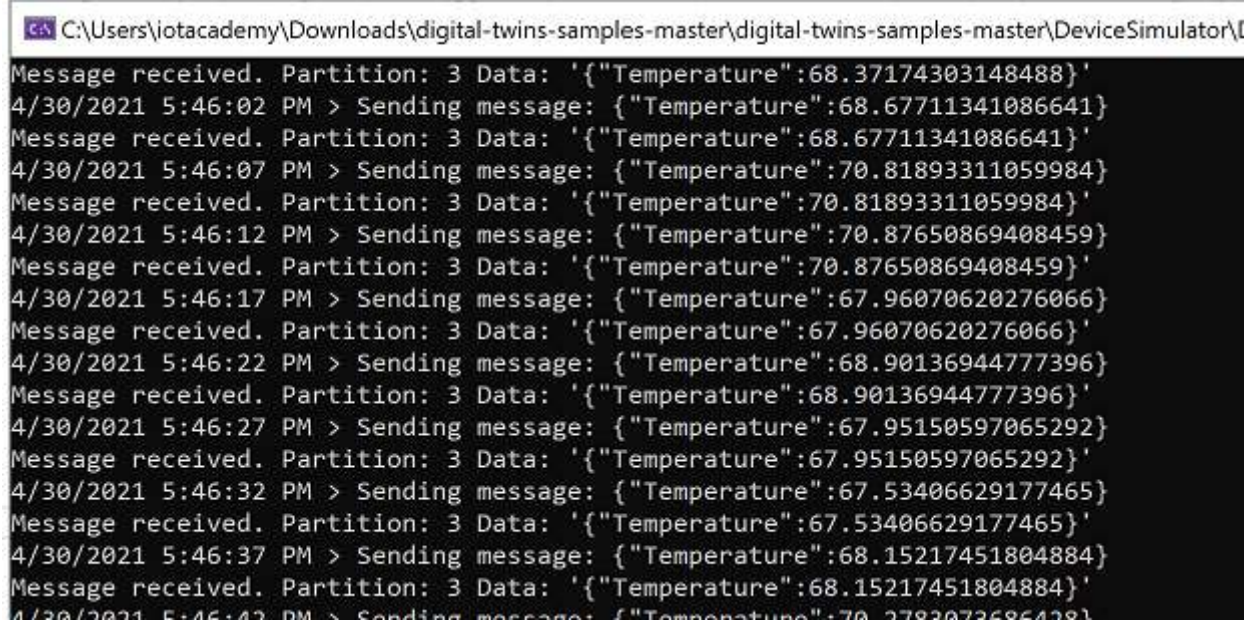
ProcessDTRoutedData [\(change\)](#)

Create

Assign a name to the event, make sure you select **Azure Function** as endpoint and then select the endpoint **ProcessDTRoutedData** from your Azure Function. Then click **Create**

2. Run the simulation and see the results Now you can run the device simulator to kick off the new event flow you've set up. Go to your Visual Studio window where the **DeviceSimulator** project is open, and run the project.

Like when you ran the device simulator earlier, a console window will open and display simulated temperature telemetry messages. These events are going through the flow you set up earlier to update the thermostat67 twin, and then going through the flow you set up recently to update the room21 twin to match.



```
C:\Users\iotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\DeviceSimulator\I
Message received. Partition: 3 Data: '{"Temperature":68.37174303148488}'
4/30/2021 5:46:02 PM > Sending message: {"Temperature":68.67711341086641}
Message received. Partition: 3 Data: '{"Temperature":68.67711341086641}'
4/30/2021 5:46:07 PM > Sending message: {"Temperature":70.81893311059984}
Message received. Partition: 3 Data: '{"Temperature":70.81893311059984}'
4/30/2021 5:46:12 PM > Sending message: {"Temperature":70.87650869408459}
Message received. Partition: 3 Data: '{"Temperature":70.87650869408459}'
4/30/2021 5:46:17 PM > Sending message: {"Temperature":67.96070620276066}
Message received. Partition: 3 Data: '{"Temperature":67.96070620276066}'
4/30/2021 5:46:22 PM > Sending message: {"Temperature":68.90136944777396}
Message received. Partition: 3 Data: '{"Temperature":68.90136944777396}'
4/30/2021 5:46:27 PM > Sending message: {"Temperature":67.95150597065292}
Message received. Partition: 3 Data: '{"Temperature":67.95150597065292}'
4/30/2021 5:46:32 PM > Sending message: {"Temperature":67.53406629177465}
Message received. Partition: 3 Data: '{"Temperature":67.53406629177465}'
4/30/2021 5:46:37 PM > Sending message: {"Temperature":68.15217451804884}
Message received. Partition: 3 Data: '{"Temperature":68.15217451804884}'
4/30/2021 5:46:42 PM > Sending message: {"Temperature":70.27820726864281}
```

To see the data from the Azure Digital Twins side, go to your Visual Studio window where the AdtE2ESample project is open, and run the project.

In the project console window that opens, run the following command to get the temperatures being reported by both the digital twin thermostat67 and the digital twin room21.

```
ObserveProperties thermostat67 Temperature room21 Temperature
```

You should see the live updated temperatures from your Azure Digital Twins instance being logged to the console every two seconds. Notice that the temperature for room21 is being updated to match the updates to thermostat67.

```
C:\Users\iotacademy\Downloads\digital-twins-samples-master\digital-twins-samples-master\Adt5
$dtId: room21, Temperature: 68.4322
$dtId: thermostat67, Temperature: 68.43219431370133
$dtId: room21, Temperature: 68.4322
$dtId: thermostat67, Temperature: 67.57605031718316
$dtId: room21, Temperature: 67.57605
$dtId: thermostat67, Temperature: 67.57605031718316
$dtId: room21, Temperature: 67.57605
$dtId: thermostat67, Temperature: 67.57605031718316
$dtId: room21, Temperature: 67.57605
$dtId: thermostat67, Temperature: 67.43707335946945
$dtId: room21, Temperature: 67.43707
$dtId: thermostat67, Temperature: 67.43707335946945
$dtId: room21, Temperature: 67.43707
$dtId: thermostat67, Temperature: 70.94210684296773
$dtId: room21, Temperature: 70.94211
$dtId: thermostat67, Temperature: 70.94210684296773
$dtId: room21, Temperature: 70.94211
$dtId: thermostat67, Temperature: 70.94210684296773
$dtId: room21, Temperature: 70.94211
$dtId: thermostat67, Temperature: 68.9105559540496
$dtId: room21, Temperature: 68.91055
```

## Exercise #4: Clean up

After completing all the exercises go to the Azure Portal and you can delete the resource group, click in **Delete resource group**

