

Azure Data Processing

LAB Overview

This lab introduces you how to build data processing using Azure services. You will use Azure Function as a metrics generator. Event hub will be main message aggregator, Stream Analytics will transmit data to Service Bus.

Task 1: Create an Event Hub namespace

In this section you will learn how to create an Event Hub namespace from Azure Portal.

1. On the left **Hub** menu click on **Create a resource**.
2. On the **New** blade click on **Internet of Things** and select **Event Hubs**.
3. On the **Create namespace** blade provide the following configurations:
 - **Name:** event-hub-student0X
 - **Pricing Tier:** Standard
 - **Enable Kafka:** None
 - **Make this namespace zero redundant:** None
 - **Subscription:** XXXXXX
 - **Resource group:** student0X
 - **Location:** West Europe
 - **Throughput Unit:** 1
 - **Enable Auto-inflate:** None

When, you finish click on button **Create**.

4. Next step the Azure Event Hub namespace will be deployed.

Task 2: Add Event Hubs to namespace

In this section, you will learn how to add a new instance to namespace of created Azure Event Hub from previous task.

1. After successful deployment of Azure Event Hub, go to created service **event-hub-studentX**. (X – your id number)
You can search the service using field search **Search resources, services and docs** which is located at the top of Azure Portal.
2. On the **Event Hubs Namespace** page click on the **Event Hub** button marked with a plus sign.
3. On the **Create Event Hub** blade provide the following configurations:
 - **Name:** dataBroker
 - **Partition Count:** 2
 - **Message Retention:** 1
 - **Capture:** Off

When you finish, click on button **Create**.

4. Next step the Event Hub will be added to namespace.
 5. Add two more event hubs in the same way: **metrics** and **events**.
-

Task 3: Add consumer groups to Event Hub

In this section, you will learn how to add consumer group to instance of Event Hub from Azure Portal.

1. On the main **Overview** page of **Event Hubs Namespace**, click on **Event Hubs** from the left menu.
2. On the **Event Hubs** page click on created instance of Event Hub - **dataBroker**.
3. On the instance of Event Hub – **dataBroker** page click on **Consumer groups** from the left menu.
4. On the **Consumer Group** page click on button **Consumer Group** marked with plus assign and add group named:
 - stream-analytics

Task 4: Create Azure Functions

In this section, you will learn how to create Azure Functions from Azure Portal.

1. Open Azure Portal and Click **Create Resource** on left menu.
 2. In Search the Marketplace write **Function App** then click **Create**.
 3. On the Create Function App page provide the following configurations:
 - **Resource Group:**
 - **Use existing:** <Your Resource Group>
 - **Function App name:** function-x-app (X – your id number)
 - **Publish:** Code
 - **Runtime stack:** .NET (X – your id number)
 - **Version:** 3.1
 - **Region:** West Europe
 - Next click on **Review + Create** and then **Create**.
-

Task 5: Create an instance of function

In this section, you will learn how to create an instance of function triggered by timer and how to add custom definition of code.

1. On the Function App page click **Functions** on the left menu and then click **Add** marked with plus assign.
2. Click on **Time trigger** method.
3. On the **Time Trigger New Function** blade provide the following configurations:
 - **Name:** MessageGeneerator
 - **Schedule:** */5 * * * * *

And click on **Create**.

4. On the **MessageGeneerator** page function go to **Code + Test** and change code in **run.csx** file:

```
#r "Newtonsoft.Json"
using System;
using Newtonsoft.Json;
public static string Run(TimerInfo myTimer, TraceWriter log)
{
```

```
log.Info($"C# Timer trigger function executed at: {DateTime.Now}");
Random value = new Random();
string json = JsonConvert.SerializeObject( new { deviceId= value.Next(0,10), a=
value.Next(0,1000), b= value.Next(0,100), Time = DateTime.Now});
log.Info($"JSON output: {json}");
return json;
}
```

And click on **Save**.

Task 6: Add Event Hub output to Function

In this section, you will learn how to add output from Event Hub to Azure Functions.

1. On the **MessageGenerator** page function click on **Integration**.
2. On the **Integrate** page click on **New Output** and select **Azure Event Hubs**.
3. Next on **Azure Event Hubs Output** section provide the following configurations:
 - **Binding Type:** Azure Event Hubs
 - **Event parameter name:** \$return
 - **Event Hub connection:**
 - **Select** Event Hub
 - **Namespace:** event-hub-student0X
 - **Event Hub:** databroker
 - **Policy:** RootMangeSharedAccessKeyAnd click on **Select**.
 - **Event Hub name:** databroker

And click on **OK**.

Task 7: Create Azure Service Bus

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Integration** → **Service Bus**.
4. On the Create container instance page provide the following configurations:
 - **Resource Group:**
 - **Use existing:** <Your Resource Group>
 - **Namespace name:** service-X-bus (X – your id number)

- **Location:** West Europe
 - **Pricing Tier:** Standard
 - Next click on **Review + create** and then **Create**
-

Task 8: Add queue to Azure Service Bus

1. On the main **Overview** page of created Azure Service Bus click on **Queues** from the left menu.
2. Next click on **Queue** button marked with plus assign.
3. On the **Create queue** blade provide the following configurations:
 - **Name:** metrics
 - **Max size:** 1 GB
 - **Max delivery count:** 10
 - **Message time to live (default):** 14 days
 - **Lock duration:** 30 seconds

And select only **Enable partitioning**.

When you finish click on **Create**.

Task 8: Create Stream Analytics

In this section you will learn how to create Azure Stream Analytics.

1. Open Azure Portal and Click **Create Resource** on left menu.
2. In Search the Marketplace write **Stream Analytics job** then click **Create**.
3. On the **New Stream Analytics job** blade provide the following configurations:
 - **Job name:** student-x-sa (X – your id number)
 - **Resource Group:**
 - **Use existing:** <Your Resource Group>
 - **Location:** West Europe
 - **Hosting Environment:** Cloud
 - **Streaming units:** 1

When, you finish click on button **Create**.

4. Next step the Azure Stream Analytics will be deployed.


Task 9: Add Event Hub input to Stream Analytics

In this section you will learn how to add input source of Event Hub in Stream Analytics.

1. On the main **Overview** page of **Stream Analytics**, click on **Inputs**.
2. On the **Inputs** page click on button **Add Stream input** and select **Event Hub**.
3. On the **Event Hub new input** provide the following configurations:

- **Input alias:** EventHub
- **Select Event Hub from your subscription**
- **Event Hub namespace:** event-hub-student0X
- **Event Hub name**
 - **Use existing:** databroker
- **Event Hub consumer group:** stream-analytics
- **Authentications mode:** Connection string
- **Event Hub policy name:** RootManageSharedAccessKey
- **Event serialization format:** JSON
- **Encoding:** UTF-8
- **Event compression type:** None

And click on **Save**.

4. Next go on created Event Hub namespace **event-hub-studentX** from previous tasks and click on **Shared Access policies**.
 5. On the **Shared Access policies** page click on **RootManageSharedAccessKey**.
 6. On the **Shared Access policies** blade copy value from **Primary key** field.
 7. Go on main main **Overview** page of **Stream Analytics**, click on **Inputs**.
 8. On **Inputs** page click on created **EventHub** input and paste value of **Primary key** from step 6 and click on **Save**.
 9. On **EventHub Input Details** click on **Test** icon ->  to check the connection with EventHub.
-

Task 10: Add Service Bus output to Stream Analytics

In this section, you will learn how to add output source of events and metrics Event Hub in Stream Analytics.

1. On the main **Overview** page of **Stream Analytics**, click on **Outputs**.
2. On the **Outputs** page click on button **Add** and select **Service Bus Queue**.
3. On the **Service Bus queue New output** provide the following configurations:
 - **Output alias:** Metrics
 - **Select Service Bus queue from your subscriptions**
 - **Service Hub namespace:** service-X-bus (X – your id number)
 - **Queue name**
 - **Use existing:** metrics
 - **Queue policy name:** RootManageSharedAccessKey
 - **Event serialization format:** JSON
 - **Encoding:** UTF-8
 - **Format:** Line separated

And click on **Save**.

4. On **ServiceBus Output Details** click on **Test** icon ->  to check the connection with ServiceBus.

Task 11: Add query to Stream Analytics

In this section you will learn how to add SQL query in Azure Stream Analytics which processing all data from Event Hub to Service Bus Queues.

1. Go to **Overview** page of **Stream Analytics**.
2. On the main **Overview** page of **Stream Analytics**, add below lines of SQL code clicking on **Edit Query** in **Query** section:

```
SELECT a, b, CAST(Time as datetime) as Time  
INTO Metrics  
FROM EventHub
```

And click on **Save query**.

3. On the main **Overview** page of **Stream Analytics** click on **Start** to run Stream Analytics job.

Task 12: Check data from Event Hub using Stream Analytics

1. Go to **Overview** page of **Stream Analytics**.
2. On the main **Overview** page of **Stream Analytics** click **Query**.
3. Click **Test query**. On the bottom of page click **Input preview** (you will see input data from Event Hub) and then click **Test result** (you will see result of your sql query).

Task 13: Check data on Service Bus queue.

1. Go to **Overview** page of **Service Bus**.
2. On the main **Overview** page of **Service Bus** click **Queues**. Then choose your queue -> **metrics**.
3. Click **Service Bus Explorer (preview)** from the left menu. Then click **Peek** section and choose **DeadLetter**.
4. Click **Peek** button. You will see a lot of messages that your queue received. Click one of them and on the right you will see window with details about your message (for example: body of your message).