

Demo 2 - F&O and Service Bus

Wednesday, October 9, 2019 3:29 PM

Requirements:

- Azure subscription
- F&O subscription
- (optional) Visual Studio 2019 or Visual Studio Code

1. Create a Service Bus namespace:

- From the Azure portal, click to **+ Create a Resource > Service Bus** namespace, set it up like this and then click **Create**:

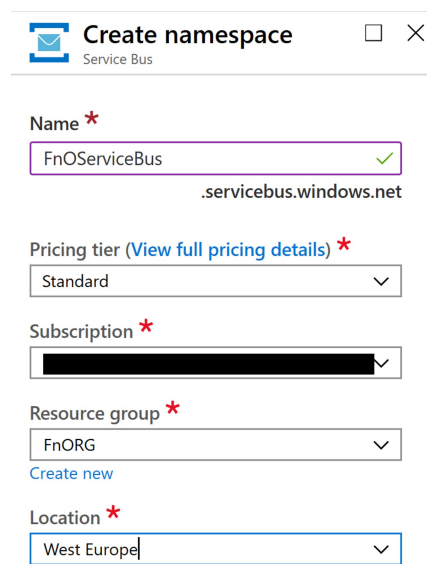
Name: give it any name

Resource group: select an existing one or create a new one

Region: West Europe

Pricing tier: Standard

Subscription: select the appropriate one

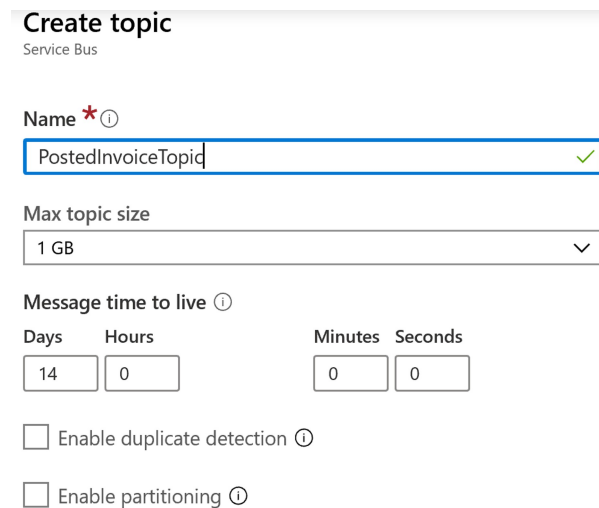


The screenshot shows the 'Create namespace' form for Service Bus. It includes fields for Name (FnOServiceBus), Pricing tier (Standard), Subscription (a redacted dropdown), Resource group (FnORG), and Location (West Europe). Each field has a red asterisk indicating it is required. There are also links for 'View full pricing details' and 'Create new' resource group.

- Once created go to **Topics > +Topic**, set it up like this and click **Create**:

Name: PostedInvoiceTopic

Leave all other values by default.



The screenshot shows the 'Create topic' form for Service Bus. It includes fields for Name (PostedInvoiceTopic), Max topic size (1 GB), and Message time to live (14 days, 0 hours, 0 minutes, 0 seconds). There are also checkboxes for 'Enable duplicate detection' and 'Enable partitioning'.

- c. Now click on the topic you just created and click on **Subscription** > **+Subscription**, set it up like this and click **Create**:

Name: PostedInvoiceTopicSubscription
Leave all other values by default.

Create subscription ☐

postedinvoicetopic

Name *

PostedInvoiceTopicSubscription ☒

Message time to live
(default) *

14 days

Lock duration *

30 ☒ seconds

max delivery count *

10

☐ Move expired messages to the dead-letter subqueue

☐ Move messages that cause filter evaluation exceptions to the dead-letter subqueue

☐ Enable sessions

- d. Go back to your service bus namespace overview and click on **Shared Access Policies** > **+Add** > give it a name and select only **Send**, then click **Create**:

Add SAS Policy

Service Bus

Policy name *

FnOServiceBusSendAccessPolicy ☒

☐ Manage

☒ Send

☐ Listen

- e. Click on the policy you just created and copy the Primary Connection String:

+ Add

policy

RootManageSharedAccessKey

FnOServiceBusSendAccessPolicy

☐ Manage

☒ Send

☐ Listen

Primary Key

Secondary Key

Primary Connection String

2. Create a new Key Vault:

- Create a new **Key Vault** resource, select the appropriate subscription and set it up like this:

Resource group: select an existing one or create a new one

Key vault name: give it any name

Region: West Europe

Pricing tier: Standard

Subscription *

Resource group * [Create new](#)

Instance details

Key vault name * ⓘ ✓

Region * ✓

Pricing tier * ⓘ ✓

- Click **Review and Create** > **Create**.

- Once created, go to the Key Vault and under **Overview** copy the **DNS Name** value:

FnKeyVaultJmoreiro
Key vault

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Delete → Move

Resource group (change)
FnORG

Location
West Europe

Subscription (change)

Subscription ID

DNS Name

Sku (Pricing tier)
Standard

Directory ID

Directory Name
Microsoft

- Under **Secrets** click on **+Generate/Import**, give this secret a name and paste the value of the Primary Connection String you copied in the previous step for the Service Bus shared access policy, leaving the rest of field values by default, then click **Create**:

Create a secret

Upload options

Manual

Name * ⓘ
ServiceBusPrimaryConnectionString ✓

Value *

Content type (optional)

Set activation date? ⓘ ☐

Set expiration date? ⓘ ☐

Enabled? ☐ Yes ☐ No

- Copy the name of the secret you just created.

3. Register a new app:

- Go to **All Services** > **Security** > **Azure Active Directory** > **App Registrations** > **New Registration**, give it a name and click **Register**.
- Once created, go to **Certificates & Secrets** > **+New Client Secret**, give the secret a name and set it to never expire:

Add a client secret

Description
FnOAppSecret

Expires
☐ In 1 year
☐ In 2 years
☒ Never

Add **Cancel**

- c. Copy the secret (make sure you copy it now, otherwise you will need to create a new one if you don't):

+ New client secret			
Description	Expires	Value	Copy to clipboard
FnOAppSecret	12/31/2299		

- d. Go to **Overview** and copy the **Application (client) ID**.

4. Add an access policy to your Key Vault:

- a. Go to the Key Vault you created earlier then go to **Access Policies** > **+Add Access Policy**, set it up like this:

Select Principal: select the App you registered earlier > **Select**
Secret permissions: select Get and list:

Secret permissions

2 selected ^

☒ Select all

Secret Management Operations

☒ Get

☒ List

- b. Click **Add**.

- c. Click **Save**.

5. Configure the business event endpoint:

- a. Go to **System Administration** > **Setup** > **Business Events** > **Business Events Catalog** > **Endpoints** > **+New**
- b. Select **Azure Service Bus Topic**, click **Next**.
- c. Give the endpoint a name and fill out all fields with the values from the previous steps, then click **Ok**:

Configure new endpoint

Endpoint name
ServiceBusTopicEndpoint

Endpoint type
Azure Service Bus Topic

Topic name
PostedInvoiceTopic

Service Bus SKU
Standard

KEY VAULT INFORMATION

Azure Active Directory application ID
[REDACTED]

Azure application secret
[REDACTED]

Key Vault DNS name
[REDACTED]

Key Vault secret name
[REDACTED]

For Azure Service Bus Topic endpoints the Key Vault secret name should be a secret containing the connection string to the Service Bus

- d. If successful, you should now see your new Service Bus Topic endpoint listed:

✓ Endpoint name ↑	Endpoint type
FlowEndpoint_{8CA6E756-9E78-4355-8616-8032C9749F8E}	Microsoft Flow
FlowEndpoint_{B271F1EC-C558-48DD-96B8-E2A605145553}	Microsoft Flow
MartinsWebsinkEndpoint	HTTPS
✓ ServiceBusTopicEndpoint	Azure Service Bus

6. Activate the business event:

- a. Go to **System Administration > Setup > Business Events > Business Events Catalog** > select the **CustFreeTextInvoicePostedBusinessEvent** event from the list (filter by Business event ID) > click on **+Activate**:

Business event catalog				Endpoints	Active events	Inactive events	Errors
+ Activate							
✓ Category	Business event ID ↑	Name					
Accounts receivable	CustFreeTextInvoicePostedBusinessEvent	Free text invoice					
Sales orders	SalesInvoicePostedBusinessEvent	Invoice					

- b. Select USMF for the legal entity and for the endpoint select the one you created in the previous step, then click **Ok**:

Configure new business event

Legal entity
USMF

The default blank value will mean all legal entities where the business event occurs.

Endpoint name
ServiceBusTopicEndpoint

- c. If you want you can go to Endpoints and confirm that the business event is now active for your endpoint:

Endpoint name ↑	Endpoint type	ServiceBusTopicEndpoint
FlowEndpoint_{8CA6E756-9E78-4355-8616-8032C9749F8E}	Microsoft Flow	Azure Service Bus Topic
FlowEndpoint_{B271F1EC-C558-48DD-96B8-E2A605145553}	Microsoft Flow	Active business events
MartinsWebsinkEndpoint	HTTPS	Category
ServiceBusTopicEndpoint	Azure Service Bus	Name
		Accounts receivable
		Free text invoice posted

7. Test your new endpoint:

- Create a free text invoice in F&O (**Accounts Receivable** > **Invoices** > **All Text Invoices** > **New**) and **Post** it.
- Go to the Service Bus you created in your Azure subscription and go to the **postedinvoicetopic** topic.
- If everything worked well, in the **Overview** on the lower part you should see 2 messages were received: the first one corresponds to a request that F&O sent out when you created the new endpoint, the other one corresponds to the actual business event (in this case, free text invoice posted):

Name	Status	message count	max delivery count	Sessions Enabled
 PostedInvoiceTopic...	Active	2	10	false

8. Create an app to consume the messages (optional):

- Open Visual Studio and **Create a New Project** of type **Console App (.Net Core)**, call it **FnOServiceBusExplorer**.
- Right click on the project > **Manage NuGet packages** > under **Browse** find **Microsoft.Azure.ServiceBus** and click **Install, Ok**, and accept the license.
- In the Program class add code to show the messages received by the Service Bus subscription for the topic, for example you can add the following code:



```
using System;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using Microsoft.Azure.ServiceBus;

namespace FnOServiceBusExplorer
{
    class Program
    {
        const string ServiceBusConnectionString = "your service bus connection string";
        const string TopicName = "postedinvoicetopic";
        const string SubscriptionName = "PostedInvoiceTopicSubscription";
        static ISubscriptionClient subscriptionClient;

        static void Main(string[] args)
        {
            MainAsync().GetAwaiter().GetResult();
        }

        static async Task MainAsync()
        {
            subscriptionClient = new SubscriptionClient(ServiceBusConnectionString, TopicName,
SubscriptionName);

            Console.WriteLine("=====");
            Console.WriteLine("Press ENTER key to exit after receiving all the messages.");
            Console.WriteLine("=====");

            RegisterOnMessageHandlerAndReceiveMessages();
            Console.ReadKey();
        }
    }
}
```

```

        await subscriptionClient.CloseAsync();
    }

    static void RegisterOnMessageHandlerAndReceiveMessages()
    {
        var messageHandlerOptions = new MessageHandlerOptions(ExceptionReceivedHandler)
        {
            MaxConcurrentCalls = 1,
            AutoComplete = false
        };

        subscriptionClient.RegisterMessageHandler(ProcessMessagesAsync,
messageHandlerOptions);
    }

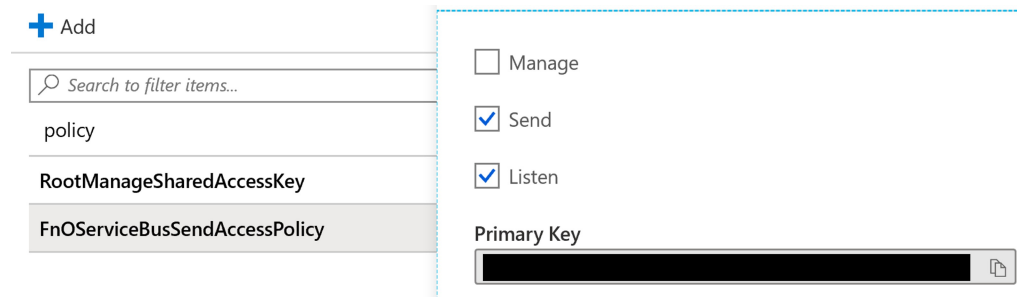
    static async Task ProcessMessagesAsync(Message message, CancellationToken token)
    {
        Console.WriteLine($"Received message:
SequenceNumber:{message.SystemProperties.SequenceNumber} Body:{Encoding.UTF8.GetString(message.Body)}");
        await subscriptionClient.CompleteAsync(message.SystemProperties.LockToken);
    }

    static Task ExceptionReceivedHandler(ExceptionReceivedEventArgs exceptionReceivedEventArgs)
    {
        Console.WriteLine($"Message handler encountered an exception
(exceptionReceivedEventArgs.Exception).");
        var context = exceptionReceivedEventArgs.ExceptionReceivedContext;
        Console.WriteLine($"Exception context for troubleshooting:");
        Console.WriteLine($"- Endpoint: {context.Endpoint}");
        Console.WriteLine($"- Entity Path: {context.EntityPath}");
        Console.WriteLine($"- Executing Action: {context.Action}");
        return Task.CompletedTask;
    }
}

```

Note: to keep things simple we added the connection string directly, however ideally you would code it to only access the connection string via the secret you created in the key vault or through environment variables / app settings.

- d. If you received an exception stating that it requires Listen claims, go to the Service Bus Shared Access Policy you created in step 1 and add the Listen:



- e. Run your app, and create a free text invoice in F&O (**Accounts Receivable > Invoices > All Text Invoices > New**) and **Post** it. In your app you should see the messages coming from the business event for the newly posted invoice:

```

C:\Program Files\dotnet\dotnet.exe
Press ENTER key to exit after receiving all the messages.
Received message: SequenceNumber:3 Body:{"BusinessEventId":"CustFreeTextInvoicePostedBusinessEvent","ControlNumber":5637145334,"EventId":"8B595511-7667-40FF-8C39-0013F5BECF88","EventTime":"/Date(1570645020000)/","InvoiceAccount":"DE-001","InvoiceAmountInAccountingCurrency":205.48,"InvoiceDate":"/Date(1570579200000)/","InvoiceDueDate":"/Date(1571443200000)/","InvoiceId":"FTI-00000030","InvoiceTaxAmount":0.0,"LegalEntity":"usmf","MajorVersion":0,"MinorVersion":0}

```