



# **PowerApps Ox Build Tools**

## **Configure Service Bus EndPoint Task**

### **Getting Started Guide**

# Getting Started Overview

This guide will demonstrate how to use the Configure Service EndPoint Task to automatically set Service EndPoint SAS parameters in the Azure DevOps Release Pipeline.

This guide assumes you have some development experience with PowerApps / Dynamics 365 and Azure Function Apps.

## Prerequisites

Please download and install Dynamics 365 Tools via nuget. This will include the Plugin Registration Tool. Please download and install Azure Service Bus Explorer (via Chocolatey).

Please follow the PowerApps Build Tools lab:

[https://github.com/microsoft/PowerApps-Samples/blob/master/azure/build-tools/PowerApps\\_Build%20Tools\\_Lab.zip](https://github.com/microsoft/PowerApps-Samples/blob/master/azure/build-tools/PowerApps_Build%20Tools_Lab.zip)

The lab completes the following tasks:

- creates 3 PowerApps CDS Environments: dev, build and prod
- creates simple PowerApp solution with 1 custom entity "Time Off Request" and a model driven app
- installs Microsoft PowerApps BuildTools into Azure DevOps
- creates Azure DevOps pipelines:
  - Create Export from Dev
  - Build Managed Solution
  - PROD Release

## Instructions

### Create Azure Service Bus

1. Log on to Azure Portal (<https://portal.azure.com>).
2. Create a "Resource Group". (If you prefer, can reuse existing Resource Group). This allows you to specify the region.

## Create a resource group

Basics Tags Review + create

**Resource group** - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. [Learn more](#)

### Project details

Subscription \* ⓘ

Free Trial

Resource group \* ⓘ

rg-demo-ox-tools-uscen

### Resource details

Region \* ⓘ

(US) Central US

3. Create two new Azure Service Busses / Queues, for dev and prod environments.

In the Azure Portal, create a new Service Bus as shown below. Note name has “dev” to indicate dev environment.

Home > Service Bus > Create namespace

### Create namespace

Service Bus

Name \*

sb-demo-ox-service-endpoint-dev-uscen✓  
.servicebus.windows.net

Pricing tier (View full pricing details) \*

Basic

Subscription \*

Free Trial

Resource group \*

rg-demo-ox-tools-uscen

Create new

Location \*

West US 2

4. Open the Service Bus and create a new Queue (under Entities). The name will be "TimeOffRequestUpdate". Example below.

### Create queue

×

Service Bus

**Name** \* ⓘ

TimeOffRequestUpdate ✓

**Max queue size**

1 GB ▾

**Message time to live** ⓘ

Days	Hours	Minutes	Seconds
14	0	0	0

**Lock duration** ⓘ

Days	Hours	Minutes	Seconds
0	0	0	30

☐ Enable duplicate detection ⓘ

☐ Enable dead lettering on message expiration ⓘ

☐ Enable sessions ⓘ

☐ Enable partitioning ⓘ

Create

5. From the Service Bus, create a new "Shared access policy" (under Security).

Ensure Send and Listen are checked.

Add SAS Policy

Service Bus

Policy name \*

PowerAppsSharedAccessKey

☐ Manage

☒ Send

☒ Listen

6. Open the newly created Share Access Policy and copy the Primary Connection String.

SAS Policy: PowerAppsSharedAccessKey

Save Discard Delete ...

☐ Manage

☒ Send

☒ Listen


Primary Key

hAv4UaoyrINyqQ+l68COqj9jIMdVYneBIT/3mAVW...


Secondary Key

V1cEhK435isfP4twlEzFyJ0Vc/zoxV7VGUO4kmIF6z...

Primary Connection String

Endpoint=sb://sb-demo-ox-service-endpoint-de... 

Secondary Connection String

Endpoint=sb://sb-demo-ox-service-endpoint-de... 

7. Repeat steps 3-6 to create “prod” Service Bus / Queue. Copy the “prod” Primary Connection String as well.

## Create PowerApps Service EndPoint

1. Register the Service EndPoint in the PowerApps environment. Open the Plugin Registration Tool → Connect to DEV environment → Register New Service Endpoint.

Enter the Primary Connection String from previous step.

Let's Start with the connection string from the Azure Service Bus Portal...

I don't have a connection string...

Next Cancel

Enter

information as shown below. Ensure Message Format is JSON.

Service Endpoint Registration

Configure a connection to Azure Service Bus to which plug-in events can be sent.

Name SB-TimeOffRequestUpdate

Namespace Address sb://sb-demo-ox-service-endpoint-dev-uscen

Designation Type Queue

Queue Name timeoffrequestupdate

Message Format JSON

Authorization Type SASKey

SAS Key Name PowerAppsSharedAccessKey

SAS Key .....

User Information Sent None

Description

Save Cancel

2. Register a New Step on the Service EndPoint to activate it. Ensure Execution Mode is Asynchronous.

## Register New Step

### General Configuration Information

Message	Update	
Primary Entity	[REDACTED]_timeoffrequest	
Secondary Entity		
Filtering Attributes	All Attributes	...
Event Handler	(ServiceEndpoint) SB-TimeOffRequestUpdate	
Step Name	SB-TimeOffRequestUpdate: Update of [REDACTED]_timeoffrequest	
Run in User's Context	Calling User	
Execution Order	1	
Description	SB-TimeOffRequestUpdate: Update of [REDACTED]_timeoffrequest	

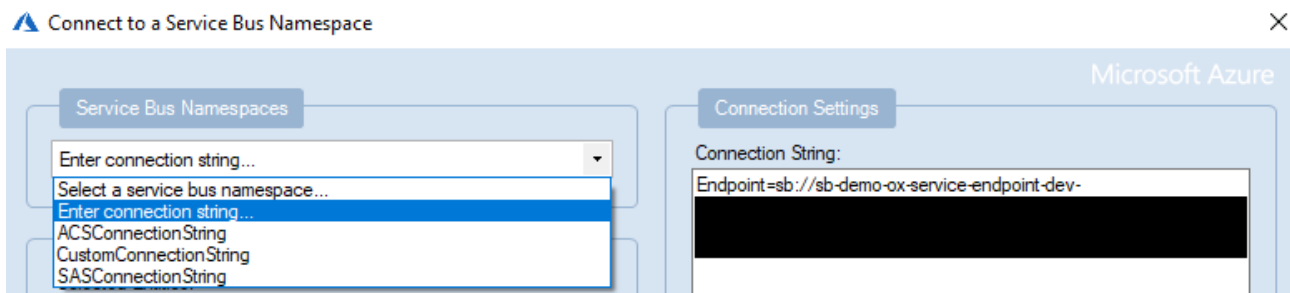
Event Pipeline Stage of Execution	Execution Mode	Deployment
PostOperation	<input checked="" type="radio"/> Asynchronous	<input checked="" type="checkbox"/> Server
	<input type="radio"/> Synchronous	<input type="checkbox"/> Offline

☒ Delete AsyncOperation if StatusCode = Successful

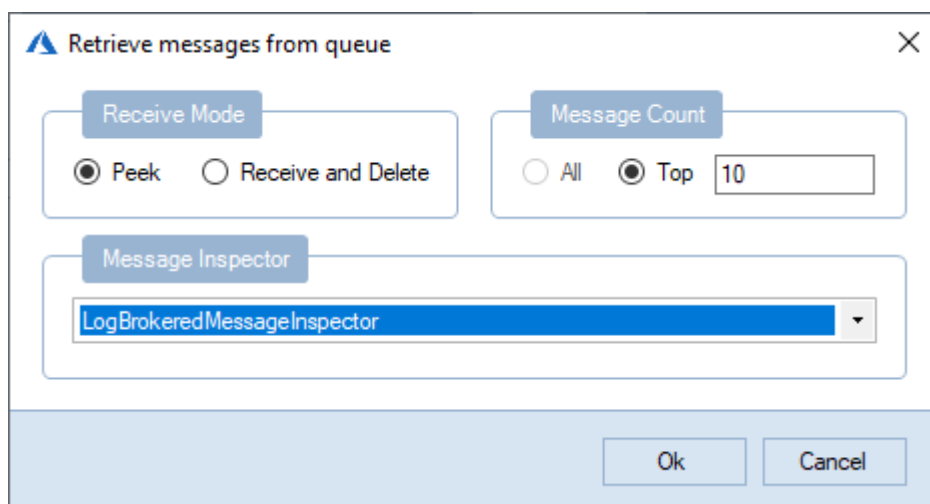
3. Test the new Service EndPoint. Log in to the PowerApps DEV environment and update a Time Off Request record.

4. Start the Service Bus Explorer. Connect to a Service Bus Namespace.

Do not use the PowerApps Connection String from previous steps. Use the default RootManageSharedAccessKey that was created automatically and has the "Manage" claim.



5. Click on the “timeoffrequestupdate” queue → Messages.

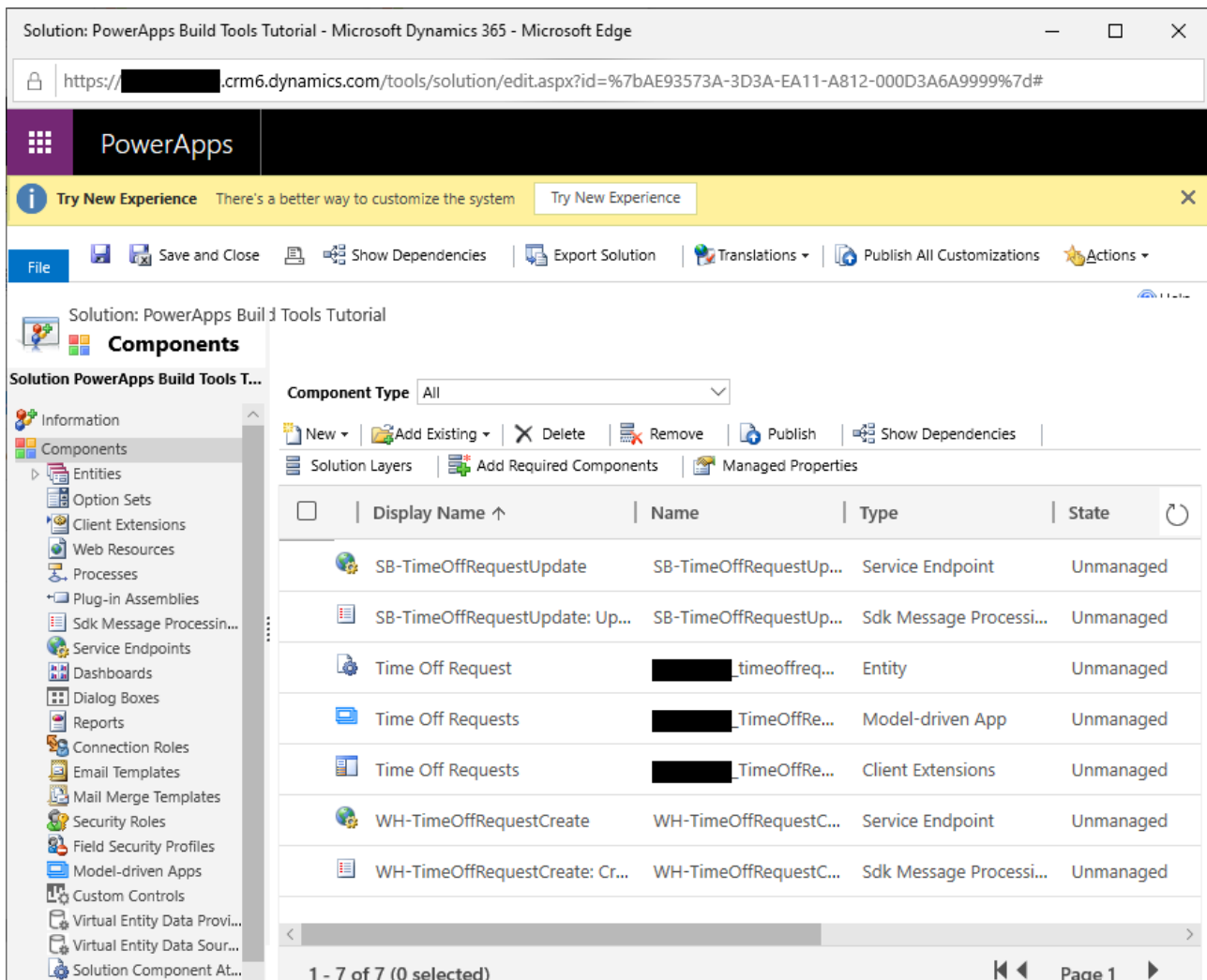


On next screen the update message will be displayed in JSON format.

5. Add the new service endpoint to the PowerApps Solution.

Open the solution → Sdk Message Processing → Add Existing  
 Select “Yes, include required components”.  
 The Solution should look similar to below.





6. Deploy the solution to build and prod environments using Azure DevOps.

Create Export from Dev  
Build Managed Solution  
PROD Release

7. Log on to the Prod PowerApps environment, open the solution, and confirm the service endpoint has been deployed to Prod. Obviously at this time, it does not have it's settings correctly configured.

## Deploy PowerApps Ox Build Tools – Configure Service Bus EndPoint

1. In Azure DevOps, go to “Browse Marketplace” and install the “PowerApps Ox Build Tools” extension.

2. Go to the Project → Pipelines → Library.

3. Create a variable group “PROD-SB-TimeOffRequestUpdate”.

Reference the PROD Primary Connection String that was copied earlier.

The 4 variables are: sepName, sepUrl, sepSASKeyName and sepSASKey

sepName	SB-TimeOffRequestUpdate (same for all environments)
sepUrl	URL from the PROD Primary Connection String
sepSASKeyName	SharedAccessKeyName from the PROD Primary Connection String
sepSASKey	SharedAccessKey from the PROD Primary Connection String

Please note that sepUrl includes the “sb://” protocol identifier, and “sepSASKey” should be a secret variable.

The Variable Group should look like below.

Library > PROD-SB-TimeOffRequestUpdate

Variable group | Save Clone Security Help

Description

☒ Allow access to all pipelines

☐ Link secrets from an Azure key vault as variables ⓘ

Variables

Name ↑	Value	🔒
sepName	SB-TimeOffRequestUpdate	
sepSASKey	*****	
sepSASKeyName	PowerAppsSharedAccessKey	
sepUrl	sb://sb-de[REDACTED]servicebus.windows.net/	

4. Edit the PROD Release Pipeline.

Link the variable group “PROD-SB-TimeOffRequestUpdate”.

Add the task “PowerApps Configure Service Bus EndPoint”.

Configure the parameters as shown below and save.

**Prod**

Deployment process

**Agent job**

Run on agent

**PowerApps Tool Installer**  
PREVIEW PowerApps Tool Installer**PowerApps Import Solution**  
PREVIEW PowerApps Import Solution**PowerApps Configure WebHook**  
PowerApps Configure WebHook**PowerApps Configure Service Bus EndPoint**  
PowerApps Configure Service Bus EndPoint

Display name \*

PowerApps Configure Service Bus EndPoint

Source ^

PowerApps Environment URL \* ⓘ | Manage

env-tut-pbuild-prod

Service EndPoint Name \* ⓘ

\$(sepName)

Service End Point URL \* ⓘ

\$(sepUrl)

Service End Point SAS Key Name \* ⓘ

\$(sepSASKeyName)

Service End Point SAS Key \* ⓘ

\$(sepSASKey)

5. Create a release and ensure it completes successfully.

6. Verify the deployment.

Log on to the PROD PowerApps environment. Update a "Time Off Request" record.

Open Service Bus Explorer and connect to the PROD service bus using the RootManageSharedAccessKey Primary Connection String. The message will be queued and can be opened and verified.

Congratulations: The Service EndPoint was configured via the Azure DevOps Release Pipeline.