100 XP

Exercise - Route custom events to web endpoint by using Azure CLI

3 minutes

In this exercise you learn how to:

- Enable an Event Grid resource provider
- Create a custom topic
- Create a message endpoint
- Subscribe to a custom topic
- Send an event to a custom topic

Prerequisites

- An **Azure account** with an active subscription. If you don't already have one, you can sign up for a free trial at https://azure.com/free .

Create a resource group

In this section, you open your terminal and create some variables that are used throughout the rest of the exercise to make command entry, and unique resource name creation, a bit easier.

1. Launch the Cloud Shell: https://shell.azure.com
2. Select **Bash** as the shell.
3. Run the following commands to create the variables. Replace `<myLocation>` with a region near you.

```
Bash
```

```
let rNum=$RANDOM*$RANDOM
myLocation=<myLocation>
myTopicName="az204-egtopic-${rNum}"
mySiteName="az204-egsite-${rNum}"
mySiteURL="https://${mySiteName}.azurewebsites.net"
```

4. Create a resource group for the new resources you're creating.

Bash

```
az group create --name az204-evgrid-rg --location $myLocation
```

Enable an Event Grid resource provider

> **Note**
>
> This step is only needed on subscriptions that haven't previously used Event Grid.

Register the Event Grid resource provider by using the `az provider register` command.

Bash

```
az provider register --namespace Microsoft.EventGrid
```

It can take a few minutes for the registration to complete. To check the status run the following command.

Bash

```
az provider show --namespace Microsoft.EventGrid --query "registration-State"
```

Create a custom topic

Create a custom topic by using the `az eventgrid topic create` command. The name

must be unique because it's part of the DNS entry.

Bash

```
az eventgrid topic create --name $myTopicName \
    --location $myLocation \
    --resource-group az204-evgrid-rg
```

Create a message endpoint

Before subscribing to the custom topic, we need to create the endpoint for the event message. Typically, the endpoint takes actions based on the event data. The following script uses a prebuilt web app that displays the event messages. The deployed solution includes an App Service plan, an App Service web app, and source code from GitHub. It also generates a unique name for the site.

1. Create a message endpoint. The deployment may take a few minutes to complete.

   Bash

   ```
   az deployment group create \
       --resource-group az204-evgrid-rg \
       --template-uri "https://raw.githubusercontent.com/Azure-Samples/azure-event-grid-viewer/main/azuredeploy.json" \
       --parameters siteName=$mySiteName hostingPlanName=viewerhost

   echo "Your web app URL: ${mySiteURL}"
   ```

   Note

   This command may take a few minutes to complete.

2. In a new tab, navigate to the URL generated at the end of the previous script to ensure the web app is running. You should see the site with no messages currently displayed.

   Tip

   Leave the browser running, it is used to show updates.

Subscribe to a custom topic

You subscribe to an Event Grid topic to tell Event Grid which events you want to track and where to send those events.

1. Subscribe to a custom topic by using the `az eventgrid event-subscription create` command. The following script grabs the needed subscription ID from your account and use in the creation of the event subscription.

```
Bash

endpoint="${mySiteURL}/api/updates"
subId=$(az account show --subscription "" | jq -r '.id')

az eventgrid event-subscription create \
    --source-resource-id "/subscriptions/$subId/resource-
Groups/az204-evgrid-rg/providers/Microsoft.EventGrid/topics/$my-
TopicName" \
    --name az204ViewerSub \
    --endpoint $endpoint
```

2. View your web app again, and notice that a subscription validation event has been sent to it. Select the eye icon to expand the event data. Event Grid sends the validation event so the endpoint can verify that it wants to receive event data. The web app includes code to validate the subscription.

Send an event to your custom topic

Trigger an event to see how Event Grid distributes the message to your endpoint.

1. Retrieve URL and key for the custom topic.

```
Bash

topicEndpoint=$(az eventgrid topic show --name $myTopicName -g
az204-evgrid-rg --query "endpoint" --output tsv)
key=$(az eventgrid topic key list --name $myTopicName -g az204-
evgrid-rg --query "key1" --output tsv)
```

2. Create event data to send. Typically, an application or Azure service would send the event data, we're creating data for the purposes of the exercise.

```bash
event='[ {"id": "'"$RANDOM"'", "eventType": "recordInserted", "subject": "myapp/vehicles/motorcycles", "eventTime": "'`date +%Y-%m-%dT%H:%M:%S%z`'", "data":{ "make": "Contoso", "model": "Monster"},"dataVersion": "1.0"} ]'
```

3. Use `curl` to send the event to the topic.

```bash
curl -X POST -H "aeg-sas-key: $key" -d "$event" $topicEndpoint
```

4. View your web app to see the event you just sent. Select the eye icon to expand the event data.

```json
{
"id": "29078",
"eventType": "recordInserted",
"subject": "myapp/vehicles/motorcycles",
"eventTime": "2019-12-02T22:23:03+00:00",
"data": {
    "make": "Contoso",
    "model": "Northwind"
},
"dataVersion": "1.0",
"metadataVersion": "1",
"topic": "/subscriptions/{subscription-id}/resourceGroups/az204-evgrid-rg/providers/Microsoft.EventGrid/topics/az204-egtopic-589377852"
}
```

Clean up resources

When you no longer need the resources in this exercise use the following command to

delete the resource group and associated resources.

Bash

```
az group delete --name az204-evgrid-rg --no-wait
```

Next unit: Knowledge check

Continue >