

Azure Event Grid is an eventing service for the cloud. In this article, you use the Azure CLI to create a custom topic, subscribe to the topic, and trigger the event to view the result. Typically, you send events to an endpoint that processes the event data and takes actions. However, to simplify this article, you send the events to a web app that collects and displays the messages.

When you're finished, you see that the event data has been sent to the web app. If you don't have an Azure subscription, create a [free account](#) before you begin.

## Open Azure Cloud Shell

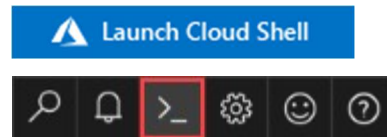
Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Just select the **Copy** button to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.



Open Cloud Shell in your browser.

Select the **Cloud Shell** button on the menu in the upper-right corner of the [Azure portal](#).



If you choose to install and use the CLI locally, this article requires that you are running the latest version of Azure CLI (2.0.24 or later). To find the version, run `az --version`. If you need to install or upgrade, see [Install Azure CLI 2.0](#).

If you aren't using Cloud Shell, you must first sign in using `az login`.

## Create a resource group

Event Grid topics are Azure resources, and must be placed in an Azure resource group. The resource group is a logical collection into which Azure resources are deployed and managed.

Create a resource group with the `az group create` command.

The following example creates a resource group named `gridResourceGroup` in the `westus2` location.

Azure CLI Copy Try It

```
az group create --name gridResourceGroup --location westus2
```

```
baejang_t@Azure:~$ az group create --name gridResourceGroup --location westus2
{
  "id": "/subscriptions/1c612728-8a3f-40cf-a96a-ea3d3c3b8fdf/resourceGroups/gridResourceGroup",
  "location": "westus2",
  "managedBy": null,
  "name": "gridResourceGroup",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

## Create a custom topic

An event grid topic provides a user-defined endpoint that you post your events to. The following example creates the custom topic in your resource group. Replace `<your-topic-name>` with a unique name for your topic. The topic name must be unique because it's part of the DNS entry.

Azure CLI Copy Try It

```
topicname=joycetopic
```

```
az eventgrid topic create --name $topicname -l westus2 -g gridResourceGroup
```

```
{
  "endpoint": "https://joycetopic.westus2-1.eventgrid.azure.net/api/events",
  "id": "/subscriptions/1c612728-8a3f-40cf-a96a-
ea3d3c3b8fdf/resourceGroups/gridResourceGroup/providers/Microsoft.EventGrid/topics/joycetopic",
  "location": "westus2",
  "name": "joycetopic",
  "provisioningState": "Succeeded",
  "resourceGroup": "gridResourceGroup",
  "tags": null,
  "type": "Microsoft.EventGrid/topics"
}
```

## Create a message endpoint

Before subscribing to the topic, let's create the endpoint for the event message. Typically, the endpoint takes actions based on the event data. To simplify this quickstart, you deploy a [pre-built 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.

Replace `<your-site-name>` with a unique name for your web app. The web app name must be unique because it's part of the DNS entry.

Azure CLI Copy Try It

siteName=joycebzhangweb

```
az group deployment create \
  --resource-group gridResourceGroup \
  --template-uri "https://raw.githubusercontent.com/dbarkol/azure-event-grid-
viewer/master/azuredploy.json" \
  --parameters siteName=$siteName hostingPlanName=viewerhost
```

The deployment may take a few minutes to complete. After the deployment has succeeded, view your web app to make sure it's running. In a web browser, navigate to: `https://<your-site-name>.azurewebsites.net`

You should see the site with no messages currently displayed.

## Subscribe to a topic

You subscribe to a topic to tell Event Grid which events you want to track and where to send those events. The following example subscribes to the topic you created, and passes the URL from your web app as the endpoint for event notification.

The endpoint for your web app must include the suffix `/api/updates/`.

Azure CLI Copy Try It

endpoint=https://\$siteName.azurewebsites.net/api/updates

```
az eventgrid event-subscription create \
  -g gridResourceGroup \
  --topic-name $topicName \
  --name demoViewerSub \
  --endpoint $endpoint
```

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.



## Azure Event Grid Viewer



Event Type	Subject
Microsoft.EventGrid.SubscriptionValidationEvent	
<pre>[{   "id": "3af3f5b7-7706-4e39-8e87-bf048efb0c60",   "topic": "/subscriptions/1c612728-8a3f-40cf-a96a-ea3d3c3b8fdf/resourceGroups/gridresourcegroup/providers/microsoft.eventgrid/topics/joycetopic",   "subject": "",   "data": {     "validationCode": "7CFFC042-8437-4C6B-B380-ADC51DC93DFB"   },   "eventType": "Microsoft.EventGrid.SubscriptionValidationEvent",   "eventTime": "2018-05-30T16:03:43.9759982Z",   "metadataVersion": "1",   "dataVersion": "2" }]</pre>	

## Send an event to your topic

Let's trigger an event to see how Event Grid distributes the message to your endpoint. First, let's get the URL and key for the custom topic.

Azure CLI Copy Try It

```
endpoint=$(az eventgrid topic show --name $topicname -g gridResourceGroup --query "endpoint" --output tsv)
key=$(az eventgrid topic key list --name $topicname -g gridResourceGroup --query "key1" --output tsv)
```

```
aejang_t@Azure:~$ endpoint=$(az eventgrid topic show --name $topicname -g gridResourceGroup --query "endpoint" --output tsv)
aejang_t@Azure:~$ key=$(az eventgrid topic key list --name $topicname -g gridResourceGroup --query "key1" --output tsv)
aejang_t@Azure:~$ ls
loudrive
aejang_t@Azure:~$ echo $endpoint
https://joycetopic.westus2-1.eventgrid.azure.net/api/events
aejang_t@Azure:~$ echo $key
msdybmquFc1ub7U+GMyakedppdZdz[REDACTED]
aejang_t@Azure:~$
```

To simplify this article, you use sample event data to send to the topic. Typically, an application or Azure service would send the event data. The following example gets the event data:

Azure CLI Copy Try It

```
body=$(eval echo "$(curl https://raw.githubusercontent.com/Azure/azure-docs-json-samples/master/event-grid/customevent.json)")
```

```
baejang_t@Azure:~$ body=$(eval echo "$(curl https://raw.githubusercontent.com/Azure/azure-docs-json-samples/master/event-grid/customevent.json)")
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left   Speed
100  305  100  305    0    0     0    846    0 --:--:-- --:--:-- --:--:--   844
baejang_t@Azure:~$ echo $body
[ { "id": "19279", "eventType": "recordInserted", "subject": "myapp/vehicles/motorcycles", "eventTime": "2018-05-30T16:09:02+0000", "data": {
  "make": "Ducati", "model": "Monster" }, "dataVersion": "1.0" } ]
baejang_t@Azure:~$
```

To see the full event, use `echo "$body"`. The `data` element of the JSON is the payload of your event. Any well-formed JSON can go in this field. You can also use the subject field for advanced routing and filtering.

CURL is a utility that sends HTTP requests. In this article, use CURL to send the event to the topic.

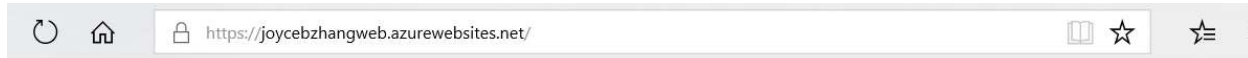
## Azure CLI Copy Try It

```
curl -X POST -H "aeg-sas-key: $key" -d "$body" $endpoint
```

You've triggered the event, and Event Grid sent the message to the endpoint you configured when subscribing. View your web app to see the event you just sent.

## JSON Copy

```
[{
  "id": "1807",
  "eventType": "recordInserted",
  "subject": "myapp/vehicles/motorcycles",
  "eventTime": "2017-08-10T21:03:07+00:00",
  "data": {
    "make": "Ducati",
    "model": "Monster"
  },
  "dataVersion": "1.0",
  "metadataVersion": "1",
  "topic": "/subscriptions/{subscription-id}/resourceGroups/{resource-group}/providers/Microsoft.EventGrid/topics/{topic}"
}]
```





## Azure Event Grid Viewer



Event Type	Subject
 recordInserted	myapp/vehicles/motorcycles
<pre>{   "id": "19279",   "eventType": "recordInserted",   "subject": "myapp/vehicles/motorcycles",   "eventTime": "2018-05-30T16:09:02+00:00",   "data": {     "make": "Ducati",     "model": "Monster"   },   "dataVersion": "1.0",   "metadataVersion": "1",   "topic": "/subscriptions/1c612728-8a3f-40cf-a96a-ea3d3c3b8fdf/resourceGroups/gridResourceGroup/providers/Microsoft.EventGrid/topics/joycetopic" }</pre>	
 Microsoft.EventGrid.SubscriptionValidationEvent	

```
make: "Ducati", "model": "Monster" }, "dataVersion": "1.0" } ]
baejang_t@Azure:~$ body=$(eval echo "$(curl https://raw.githubusercontent.com/Azure/azure-docs-json-samples/master/event-grid/customevent.json)")
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total    Spent    Left   Speed
100 305 100 305 0 0 1499 0 --:--:-- --:--:-- --:--:-- 1502
baejang_t@Azure:~$ echo $body
[ { "id": "14795", "eventType": "recordInserted", "subject": "myapp/vehicles/motorcycles", "eventTime": "2018-05-30T16:13:30+0000", "data": {
"make": "Ducati", "model": "Monster" }, "dataVersion": "1.0" } ]
baejang_t@Azure:~$ curl -X POST -H "aeg-sas-key: $key" -d "$body" $endpoint
```

Event Type	Subject
 recordInserted	myapp/vehicles/motorcycles
<pre>{   "id": "14795",   "eventType": "recordInserted",   "subject": "myapp/vehicles/motorcycles",   "eventTime": "2018-05-30T16:13:30+00:00",   "data": {     "make": "Ducati",     "model": "Monster"   },   "dataVersion": "1.0",   "metadataVersion": "1",   "topic": "/subscriptions/1c612728-8a3f-40cf-a96a-ea3d3c3b8fdf/resourceGroups/gridResourceGroup/providers/Microsoft.EventGrid/topics/joycetopic" }</pre>	
 recordInserted	myapp/vehicles/motorcycles
 recordInserted	myapp/vehicles/motorcycles

## Clean up resources

If you plan to continue working with this event or the event viewer app, don't clean up the resources created in this article. Otherwise, use the following command to delete the resources you created in this article.

Azure CLI Copy Try It

```
az group delete --name gridResourceGroup
```