## 存储队列实验

**创建资源组**

az group create --name chengzhqslab --location southeastasia

**创建存储账号**

az storage account create --name chengzhqstor0917 -g chengzhqslab --kind StorageV2 --sku Standard\_LRS

**获取存储连接字符串，记录**

az storage account show-connection-string --name chengzhqstor0917 --resource-group chengzhqslab

**创建应用**

dotnet new console -n QueueApp

**进入应用文件夹**

cd QueueApp

**构建应用**

dotnet build

**将 Microsoft.Azure.Storage.Common 和 Microsoft.Azure.Storage.Queue 包安装到项目**

dotnet add package Microsoft.Azure.Storage.Common

dotnet add package Microsoft.Azure.Storage.Queue

**打开代码编辑器**

code .

**将此前保存的存储连接字符串进行替换，**

using System;

using System.Threading.Tasks;

using Microsoft.Azure.Storage;

using Microsoft.Azure.Storage.Queue;

namespace QueueApp

{

class Program

{

// The string value is broken up for better onscreen formatting

private const string connectionString = "存储链接字符串";

static async Task Main(string[] args)

{

CloudStorageAccount storageAccount = CloudStorageAccount.Parse(connectionString);

CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();

CloudQueue queue = queueClient.GetQueueReference("mystoragequeue");

if (args.Length > 0)

{

string value = String.Join(" ", args);

await SendMessageAsync(queue, value);

Console.WriteLine($"Sent: {value}");

}

else

{

string value = await ReceiveMessageAsync(queue);

Console.WriteLine($"Received {value}");

}

Console.Write("Press Enter...");

Console.ReadLine();

}

static async Task SendMessageAsync(CloudQueue theQueue, string newMessage)

{

bool createdQueue = await theQueue.CreateIfNotExistsAsync();

if (createdQueue)

{

Console.WriteLine("The queue was created.");

}

CloudQueueMessage message = new CloudQueueMessage(newMessage);

await theQueue.AddMessageAsync(message);

}

static async Task<string> ReceiveMessageAsync(CloudQueue theQueue)

{

bool exists = await theQueue.ExistsAsync();

if (exists)

{

CloudQueueMessage retrievedMessage = await theQueue.GetMessageAsync();

if (retrievedMessage != null)

{

string theMessage = retrievedMessage.AsString;

await theQueue.DeleteMessageAsync(retrievedMessage);

return theMessage;

}

else

{

Console.Write("The queue is empty. Attempt to delete it? (Y/N) ");

string response = Console.ReadLine();

if (response == "Y" || response == "y")

{

await theQueue.DeleteIfExistsAsync();

return "The queue was deleted.";

}

else

{

return "The queue was not deleted.";

}

}

}

else

{

return "The queue does not exist. Add a message to the command line to create the queue and store the message.";

}

}

}

}

将program.cs现有代码清空，贴入这段代码，并保存

**生成项目**

dotnet build

**运行以下命令将第一个消息添加到队列**

dotnet run First queue message

**消费消息**

dotnet run

**反复几次查看消息队列的运行机制**

**可以到portal中查看队列相关信息**

**环境清理**

az group delete -n chengzhqslab --yes

代码详细说明请见：

<https://docs.microsoft.com/zh-cn/azure/storage/queues/storage-tutorial-queues?toc=%2fazure%2fstorage%2fqueues%2ftoc.json>

## ServiceBus 实验

[**准备环境**](https://docs.microsoft.com/zh-cn/learn/modules/implement-message-workflows-with-service-bus/3-exercise-implement-a-service-bus-topic-and-queue)

* 资源组：ServiceBusLab
* 服务总线命名空间：chengzhservicebus
* 定价级别：标准
* 队列名称：salesmessages
* 主题名称：salesperformancemessages
* 订阅名称：Americas

EuropeAndAfrica

**克隆项目**

cd ~

git clone <https://github.com/MicrosoftDocs/mslearn-connect-services-together.git>

（备选地址：https://github.com/cloudzun/mslearn-connect-services-together.git ）

**打开编辑器**

cd mslearn-connect-services-together/implement-message-workflows-with-service-bus/src/start

code . （常规模式）

cd mslearn-connect-services-together/implement-message-workflows-with-service-bus/src/final

code . （最简步骤）

[**队列实验**](https://docs.microsoft.com/zh-cn/learn/modules/implement-message-workflows-with-service-bus/5-exercise-write-code-that-uses-service-bus-queues)

**查看连接字符串**

az servicebus namespace authorization-rule keys list \

--resource-group ServiceBusLab \

--name RootManageSharedAccessKey \

--query primaryConnectionString \

--output tsv \

--namespace-name chengzhservicebus

**编辑privatemessagesender/Program.cs（最简模式仅需要更新连接字符串）**

**向队列发送消息**

dotnet run -p privatemessagesender

**查看消息**

az servicebus queue show \

--resource-group ServiceBusLab \

--name salesmessages \

--query messageCount \

--namespace-name chengzhservicebus

**编辑privatemessagereceiver/Program.cs（最简模式仅需要更新连接字符串）**

**从队列检索消息**

dotnet run -p privatemessagereceiver

**查看消息**

az servicebus queue show \

--resource-group ServiceBusLab \

--name salesmessages \

--query messageCount \

--namespace-name chengzhservicebus

[**总线实验**](https://docs.microsoft.com/zh-cn/learn/modules/implement-message-workflows-with-service-bus/7-exercise-write-code-that-uses-service-bus-topics)

**编辑performancemessagesender/Program.cs（最简模式仅需要更新连接字符串）**

**向主题发送消息**

dotnet run -p performancemessagesender

**查看订阅消息数**

az servicebus topic subscription show \

--resource-group ServiceBusLab \

--namespace-name chengzhservicebus \

--topic-name salesperformancemessages \

--name Americas \

--query messageCount

**编辑performancemessagereceiver/Program.cs（最简模式仅需要更新连接字符串）**

**向主题检索消息**

dotnet run -p performancemessagereceiver

**查看订阅消息数**

az servicebus topic subscription show \

--resource-group ServiceBusLab \

--namespace-name chengzhservicebus \

--topic-name salesperformancemessages \

--name Americas \

--query messageCount

## Event Hub 实验

**创建实验资源**

**创建资源组**

az group create --name learn-big-data-messaging-rg --location southeastasia

**设定后续实验默认值**

az configure --defaults group=learn-big-data-messaging-rg location=southeastasia

**定义名称空间环境变量**

NS\_NAME=chengzhns0917

**创建名称空间**

az eventhubs namespace create --name $NS\_NAME

**提取事件中心命名空间的连接字符串，并记录**

az eventhubs namespace authorization-rule keys list --name RootManageSharedAccessKey --namespace-name $NS\_NAME

记录事件中心命名空间的 primaryKey 密钥的值。

**定义事件中心环境变量**

HUB\_NAME=chengzhhub0917

**创建eventhub**

az eventhubs eventhub create --name $HUB\_NAME --namespace-name $NS\_NAME

**查看evenhub详细信息**

az eventhubs eventhub show --namespace-name $NS\_NAME --name $HUB\_NAME

**定义存储账号环境变量**

STORAGE\_NAME=chengzhehstor0917

**创建存储账号**

az storage account create --name $STORAGE\_NAME --sku Standard\_RAGRS --encryption blob

**列出访问密钥**

az storage account keys list --account-name $STORAGE\_NAME

**列出连接字符串信息，并记录**

az storage account show-connection-string -n $STORAGE\_NAME

记录存储帐户连接字符

**创建一个“messages”容器，需要使用上一步骤的连接字符串**

az storage container create -n messages --connection-string "<connection string here>"

**创建应用程序**

**克隆代码库**

cd ~

git clone <https://github.com/Azure/azure-event-hubs.git>

备用地址 :<https://github.com/cloudzun/azure-event-hubs.git>

**进入SimpleSend文件夹并打开编辑器**

cd azure-event-hubs/samples/Java/Basic/SimpleSend/src/main/java/com/microsoft/azure/eventhubs/samples/SimpleSend

code .

**替换字符串**

"Your Event Hubs namespace name" 替换为 chengzhns0917。

"Your Event Hub" 替换为 chengzhhub0917。

"Your policy name" 替换为 RootManageSharedAccessKey。

"Your primary SAS key" 替换为适用于先前所保存的事件中心命名空间的“primaryKey”密钥的值。

**切换目录**

cd ~/azure-event-hubs/samples/Java/Basic/SimpleSend

**生成 Java SimpleSend 应用程序**

mvn clean package -DskipTests

**切换到EventProcessorSample 文件夹并打开代码编辑器**

cd ~/azure-event-hubs/samples/Java/Basic/EventProcessorSample/src/main/java/com/microsoft/azure/eventhubs/samples/eventprocessorsample

code .

**替换字符串**

----ServiceBusNamespaceName----替换为 chengzhns0917 。

----EventHubName---- 替换为 chengzhhub0917。

----SharedAccessSignatureKeyName---- 替换为 RootManageSharedAccessKey。

----SharedAccessSignatureKey---- 替换为适用于先前所保存的事件中心命名空间的 primaryKey 密钥的值。

----AzureStorageConnectionString---- 替换为先前所保存的存储帐户连接字符串。

----StorageContainerName----替换为 messages。

----HostNamePrefix---- 替换为 chengzhehstor0917

**切换目录**

cd ~/azure-event-hubs/samples/Java/Basic/EventProcessorSample

**生成 Java SimpleSend 应用程序**

mvn clean package -DskipTests

**启动SipmleSend**

cd ~/azure-event-hubs/samples/Java/Basic/SimpleSend

java -jar ./target/simplesend-1.0.0-jar-with-dependencies.jar

**启动 EventProcessorSample 应用程序**

cd ~/azure-event-hubs/samples/Java/Basic/EventProcessorSample

java -jar ./target/eventprocessorsample-1.0.0-jar-with-dependencies.jar

[**监控事件中心**](https://docs.microsoft.com/zh-cn/learn/modules/enable-reliable-messaging-for-big-data-apps-using-event-hubs/7-exercise-evaluate-the-performance-of-the-deployed-event-hub-using-the-azure-portal)

**从portal中查看传入和传出请求数值**

**启动事件传入**

cd ~

cd azure-event-hubs/samples/Java/Basic/SimpleSend

java -jar ./target/simplesend-1.0.0-jar-with-dependencies.jar

**禁用事件中心状态**

**五分钟后，启用事件中心状态并启动事件传出**

cd ~

cd azure-event-hubs/samples/Java/Basic/EventProcessorSample

java -jar ./target/eventprocessorsample-1.0.0-jar-with-dependencies.jar

你将看到，虽然消息是在事件中心脱机一段时间之前发送的，但所有 100 条消息都已成功传输。