using System;

using System.Threading.Tasks;

using Azure.Messaging.ServiceBus;

namespace MessagePublisher

{

    public class Program

    {

        /\* The `<serviceBus-connection-string>` placeholder represents

           the connection string to the target Azure Service Bus namespace \*/

        private const string serviceBusConnectionString = "Endpoint=sb://sbnamespace46767742.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=mrFlpOS1874I0Xf7o26wRtVou48hjKTXe+ASbEObzng=";

        /\* To create a string constant named "queueName" with a value

           of "messagequeue", matching the name of the Service Bus queue.\*/

        private const string queueName = "messagequeue";

        /\* Stores the number of messages to be sent to the target queue \*/

        private const int numOfMessages = 3;

        /\* To create a Service Bus client that will own the connection to the target queue \*/

        static ServiceBusClient client = default!;

        /\* To create a Service Bus sender that will be

           used to publish messages to the target queue \*/

        static ServiceBusSender sender = default!;

        public static async Task Main(string[] args)

        {

            /\* To initialize "client" of type "ServiceBusClient" that will

               provide connectivity to the Service Bus namespace and "sender"

               that will be responsible for sending messages \*/

            client = new ServiceBusClient(serviceBusConnectionString);

            sender = client.CreateSender(queueName);

            /\* To create a "ServiceBusMessageBatch" object that will allow you to combine

               multiple messages into a batch by using the "TryAddMessage" method \*/

            using ServiceBusMessageBatch messageBatch = await sender.CreateMessageBatchAsync();

            /\* To add messages to a batch and throw an exception if a message

               size exceeds the limits supported by the batch \*/

            for (int i = 1; i <= numOfMessages; i++)

            {

                if (!messageBatch.TryAddMessage(new ServiceBusMessage($"Message {i}")))

                {

                    throw new Exception($"The message {i} is too large to fit in the batch.");

                }

            }

            try

            {

                /\* To create a try block, with "sender" asynchronously

                   publishing messages in the batch to the target queue \*/

                await sender.SendMessagesAsync(messageBatch);

                Console.WriteLine($"A batch of {numOfMessages} messages has been published to the queue.");

            }

            finally

            {

                /\* To create a finally block that asynchronously disposes of the "sender"

                   and "client" objects, releasing any network and unmanaged resources \*/

                await sender.DisposeAsync();

                await client.DisposeAsync();

            }

        }

    }

}