Getting Started with Azure Messaging using Azure Service Bus

Ajay Pathak
https://www.linkedin.com/in/pathakajay/
https://github.com/ajaypathak
https://community.gartner.com/u/ajay_pathak/summary

Personal Information

2007

B Tech Computer Science and Engineering From Maulana Azad National Institute of Technology, Bhopal

2007

Software Developer

CSC

2011

Technical Consultant

2012

SDE III

talentica

SPREAD YOUR WINGS

2013

Solution Architect

Schlumberger

Agenda

- What is Message
- Messaging Options in Azure
- What is Azure Service Bus
- Azure Service Bus V/S Storage Queue
- Azure Service Bus Demo
- Azure Service Bus Message
- Service Bus Queues
- Azure Service Bus Correlation
- Azure Service Bus – Duplicate Messages
- Azure Service Bus – Topic
- Azure Service Bus Tiers
- Performance and Reliability
- Q & A

Event streaming is not “modern” and Queues are not “traditional”

Both are patterns of state-of-the art messaging infrastructure

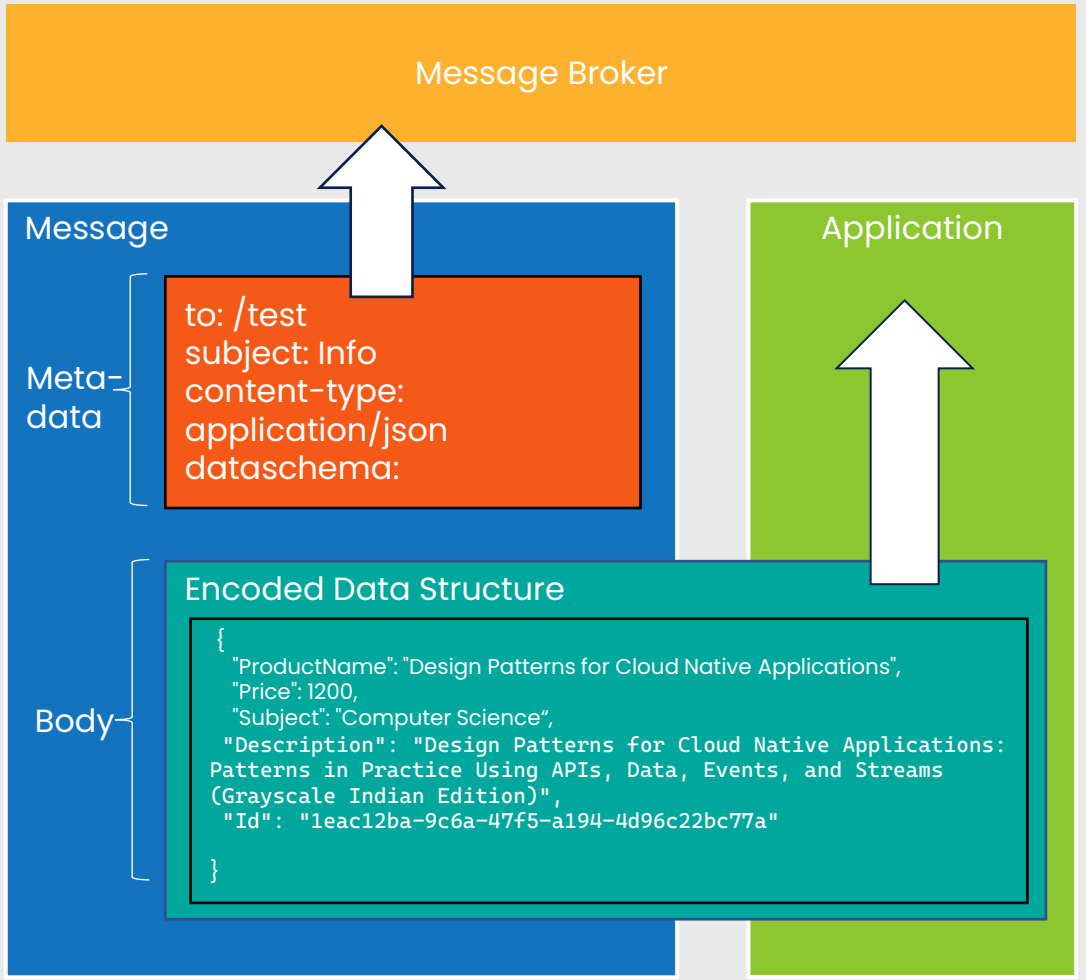
What is a Message ?

A message is an envelope annotated with metadata around a data structure to be moved between applications or services.

Data is for the apps

Metadata is for the messaging infrastructure and dispatch code.

To, Subject, Content-Type, CorrelationId, SessionId



Event vs Message

Event	Message
A particular type of message	Raw data produced by a service to be consumed or stored elsewhere
A lightweight notification	The publisher has expectation about how the message will be handled
The event data has information about what happened but doesn't have the data that triggered the event	Asynchronous operation
Producer has no expectation how event will be handled	Contract exists between two sides
The events are time-ordered and interrelated. he consumer needs the sequenced series of events to analyse what happened	Messages may or may not be related
Example : Telemetry data from IoT Devices, application logging, transaction processing	Example : Load balancing, financial transaction processing, order processing

Messaging Option in Azure

Service	Purpose	Type	When to use
Event Grid	Reactive programming	Event Distribution	React to status change
Event Hub	Big Data Pipeline	Event Streaming	Telemetry and distributed data streaming
Service Bus	High-value enterprise messaging	Message	Order processing and financial transactions

https://learn.microsoft.com/en-us/azure/event-grid/compare-messaging-services

What is Azure Service Bus?

Azure Service Bus is a fully managed enterprise message broker with message queues and publish-subscribe topics (in a namespace). Service Bus is used to decouple applications and services from each other, providing the following benefits:

- Load-balancing work across competing workers
- Safely routing and transferring data and control across service and application boundaries
- Coordinating transactional work that requires a high-degree of reliability

Azure Service Bus

- Point to point communication
- Service bus queues
  - Point to point communication
- Service bus topics
  - Point to multipoint communication
  - Filtering and actions
- Service bus sessions
- Duplicate Detection
- Scheduled Messages
- Message Deferral
- Service bus relay
- Dead-lettering
- Transactions

Azure Service Bus V/S Storage Queue

	Storage Queue	Service Bus Queue
Use Case	Basic Queue Operation	Advance queue operations (Sessions, transaction, duplicate detection, pub/sub)
Maximum queue size	500 TB	1 GB to 80 GB
Maximum Number of Queue	Unlimited	10000 per service namespace
Scheduled delivery	Yes	Yes
Automatic dead lettering	No	Yes
Server-side transaction log	Yes	No
Message auto forwarding	No	Yes
Message Groups	No	Yes (using sessions)
Duplicate detection	No	Yes
Fetching message sessions by ID	No	Yes
Atomic operation support	No	Yes
Ordering guarantee	No	Yes (FIFO) By using Sessions
Delivery guarantee	At-Least-Once	At-Least-Once (using PeekLock receive mode. It's the default) At-Most-Once (using ReceiveAndDelete receive mode)
Batched receive	Yes	Yes
Batched Send	No	Yes(by using transactions or client-side batching)

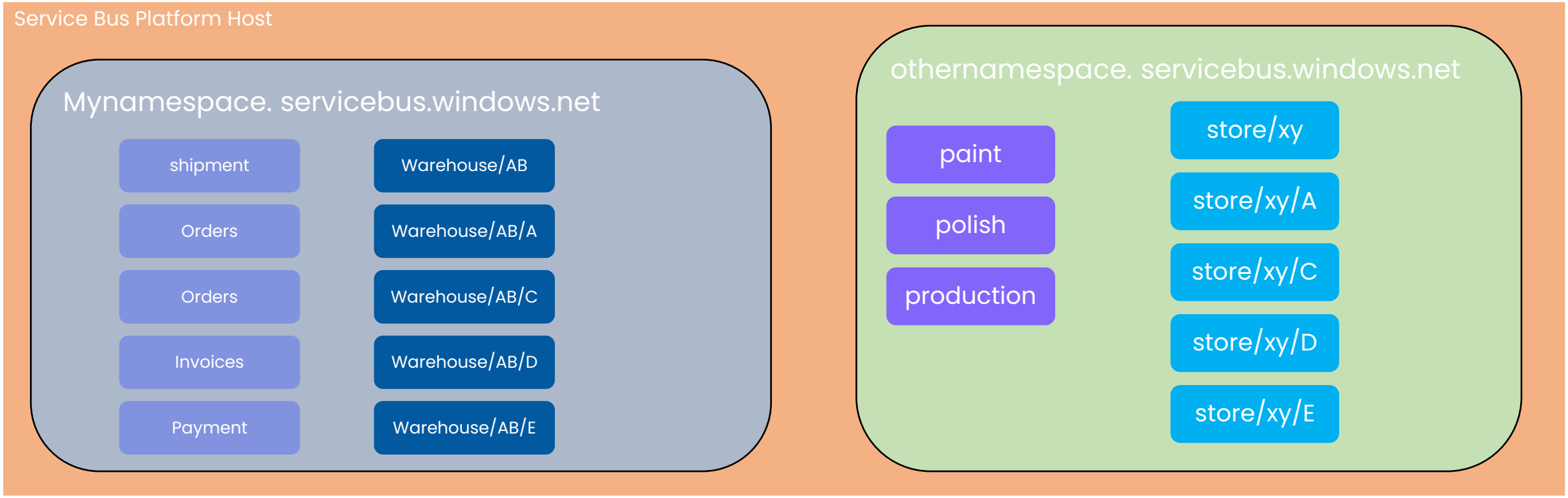
https://learn.microsoft.com/en-us/azure/service-bus-messaging/service-bus-azure-and-service-bus-queues-compared-contrasted

Service Use Cases

- Connecting on-premise LOB to cloud
- Communicating across different networks

Service Bus Namespace

A namespace is a container for all messaging components (queues and topics). Multiple queues and topics can be in a single namespace, and namespaces often serve as application containers.



Service Demo

- Creating Azure Service Bus
  - Portal
  - PowerShell

Azure CLI

```
az servicebus -h
```

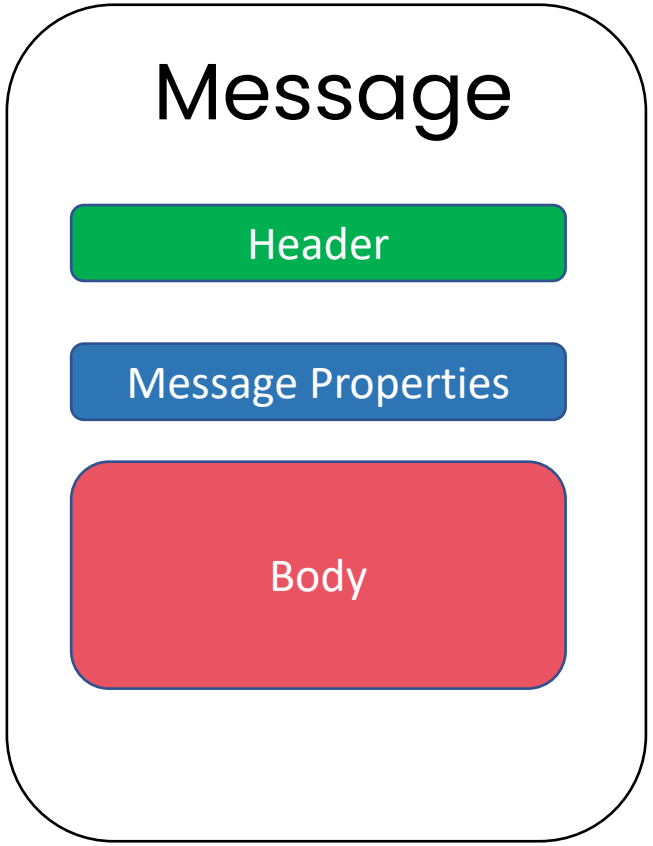
```
az servicebus namespace create --resource-group RGServiceBus --name servicebusdemo --location centralus
```

```
az servicebus queue create --name queue1 --namespace-name servicebusdemo --resource-group RGServiceBus
az servicebus queue create --name queue2 --namespace-name servicebusdemo --resource-group RGServiceBus
az servicebus queue create --name queue3 --namespace-name servicebusdemo --resource-group RGServiceBus
```

```
az servicebus topic create --name topic1 --namespace-name servicebusdemo --resource-group RGServiceBus
```

```
az servicebus topic subscription create --name subscription1 --topic-name topic1 --namespace-name servicebusdemo --resource-group RGServiceBus
az servicebus topic subscription create --name subscription2 --topic-name topic1 --namespace-name servicebusdemo --resource-group RGServiceBus
az servicebus topic subscription create --name subscription3 --topic-name topic1 --namespace-name servicebusdemo --resource-group RGServiceBus
```

Azure Service Bus Message



Message Headers and Properties

ContentType (content-type)
CorrelationId (correlation-id)
DeadLetterSource
DeliveryCount
EnqueuedSequenceNumber
EnqueuedTimeUtc
ExpiresAtUtc (absolute-expiry-time)
Label or Subject (subject)
TimeToLive
To (to)

LockedUntilUtc
LockToken
MessageId (message-id)
PartitionKey
ReplyTo (reply-to)
ReplyToSessionId (reply-to-group-id)
ScheduledEnqueueTimeUtc
SequenceNumber
SessionId (group-id)
ViaPartitionKey

```
// Copyright (c) Microsoft Corporation. All rights reserved.
// Licensed under the MIT License.

using System;

namespace Azure.Core.Amqp
{
    /// <summary>
    /// Represents the AMQP message properties.
    /// <seealso href="http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-messaging-v1.0-os.html#type-properties" />
    /// </summary>
    public class AmqpMessageProperties...
}
```

http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-messaging-v1.0-os.html#type-properties

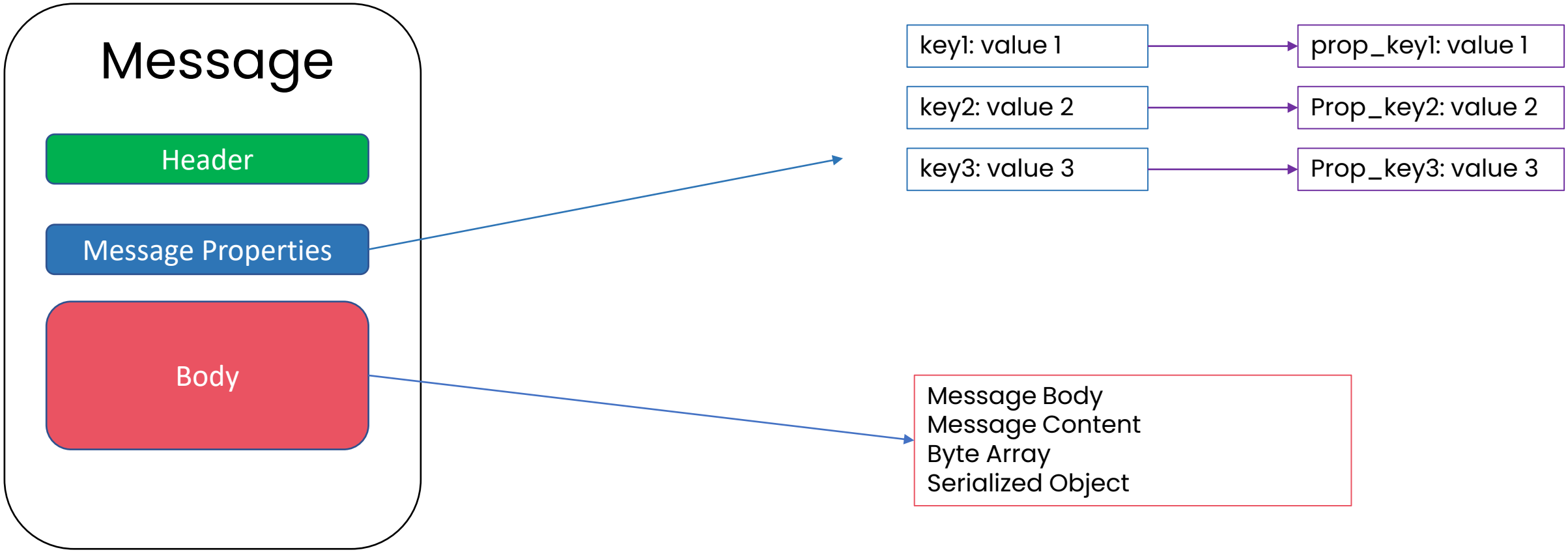
Message Headers and Properties

BrokeredMessage (SBMP) Parts	Type	HTTP header	Accessibility	HTTP Req/Res
DeliveryCount	int	BrokerProperties {DeliveryCount }	get	Res
LockedUntilUtc	DateTime	BrokerProperties{LockedUntil}	get	Res
LockToken	Guid	BrokerProperties{LockToken}	get	Res
EnqueuedTimeUtc	DateTime	Date	get	Res
SequenceNumber	long	BrokerProperties {SequenceNumber}	get	Res
TimeToLive	TimeSpan	BrokerProperties collection {TimeToLive}	get, set	Req, Res
To	string	BrokerProperties {To}	get, set	Req, Res
ScheduledEnqueueTimeUtc	DateTime	BrokerProperties {ScheduledEnqueueTimeUtc}	get, set	Req, Res
ReplyToSessionId	string	BrokerProperties {ReplyToSessionId}	get, set	Req, Res
PartitionKey	string	BrokerProperties {PartitionKey}	get, set	Req, Res
MessageId	string	BrokerProperties{MessageId}	get, set	Res
Label	string	BrokerProperties {Label}	get, set	Req, Res
ReplyTo	string	BrokerProperties {ReplyTo}	get, set	Req, Res
ContentType	string	Content-Type	get, set	Req, Res
CorrelationId	string	BrokerProperties{CorrelationId}	get, set	Req, Res
SessionID	string	BrokerProperties {SessionId}	get, set	Req, Res

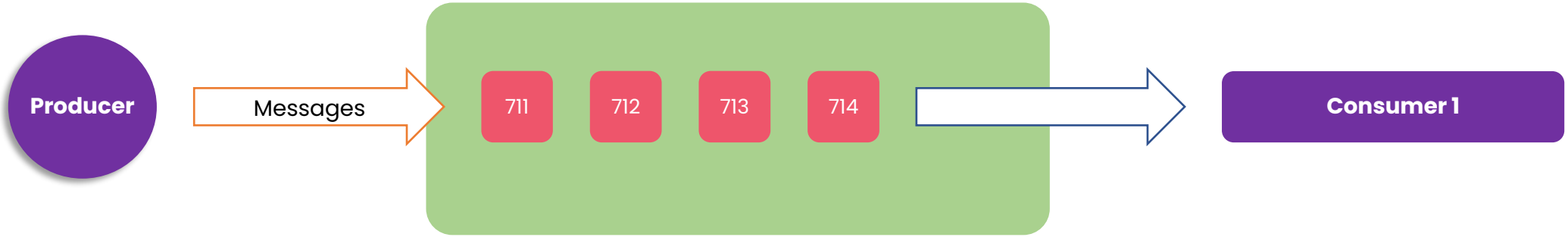
Class Name : microsoft.servicebus.messaging.brokeredmessage

https://learn.microsoft.com/en-us/rest/api/servicebus/message-headers-and-properties

Azure Service Bus Message



Service Bus Queues



Max message size:

1. Standard: **Up to 1MB**
2. Premium: **Up to 100 MB**

Max queue size: **Up to 80GB**

Every queue and topic subscription has its own **dead-letter subqueue**: myqueue/$deadletterqueue

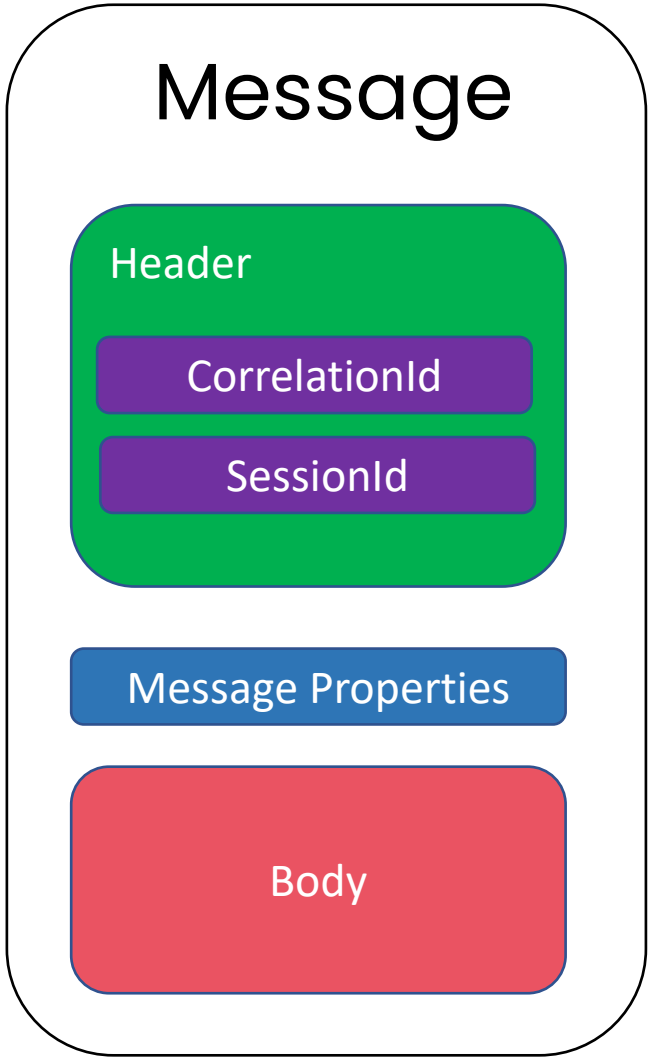
Create Queue

- Azure Portal
- PowerShell
- CLI
- Azure Resource Manager Templates
- Programming Languages

Service Bus Queues - Demo

Send Receive Messages

Azure Service Bus Correlation



Correlation Id

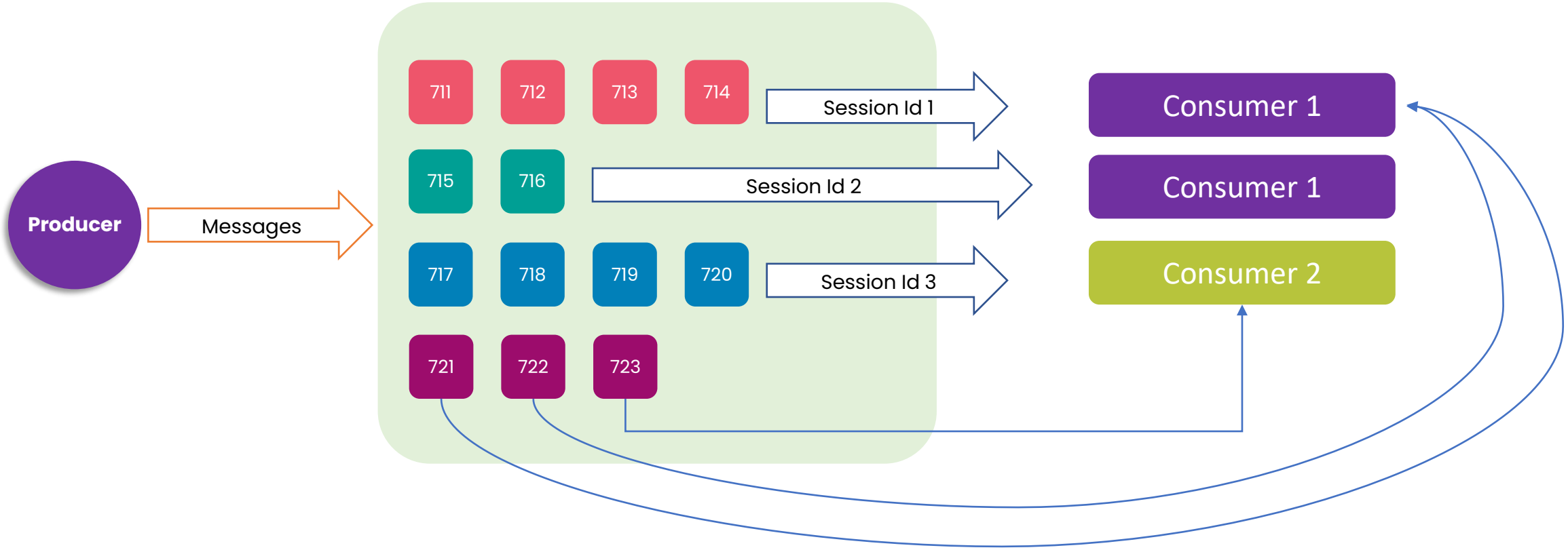
Message Routing between Topics and Subscriptions

Session Id

Correlate messages for consumer applications

Service Bus Sessions

- A logical sub queue
- A single queue can maintain hundred of thousands of queue
- Enable Sessions
- Standard and Premium Tier Support Sessions



Azure Service Bus –Sessions

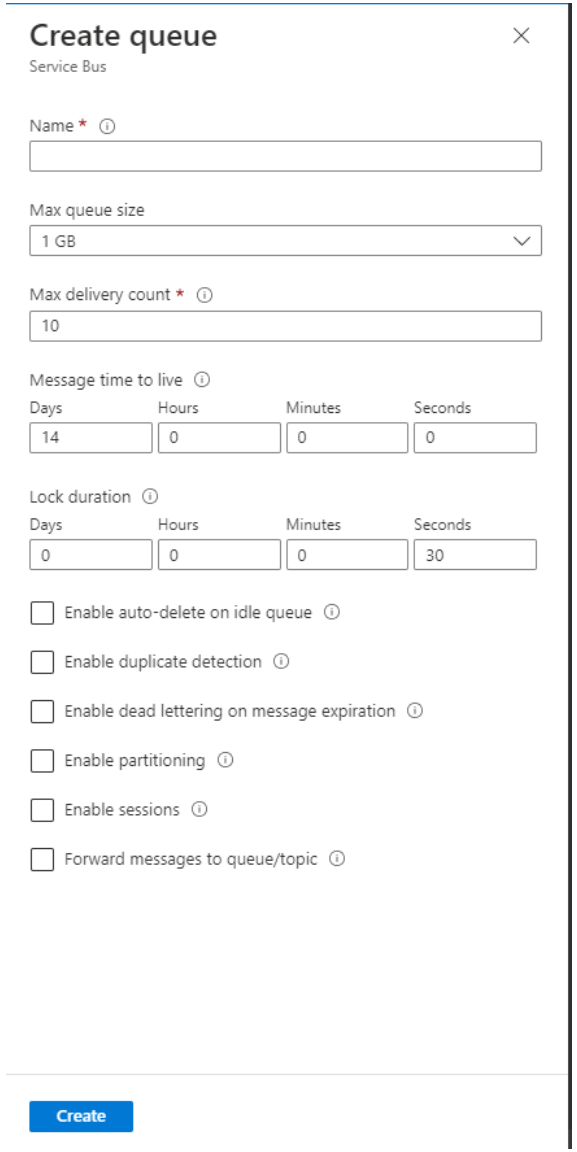
Demo

Azure Service Bus – Duplicate Messages

- Enable duplicate detection
- Duplicate Detection can be enabled for Queue or Topic
- Duplicate Detection is based on MessageId Property of ServiceBusMessage. Sender Must set MessageId
- Duplicate messages are not dead-lettered
- Standard and Premium Tier Support Sessions
- When partitioning is enabled, MessageId+PartitionKey is used to determine uniqueness. When sessions are enabled, partition key and session ID must be the same.
- When partitioning is disabled (default), only MessageId is used to determine uniqueness.

Azure Service Bus – Duplicate Messages

Portal



Azure Service Bus – Duplicate Messages

Code

```
using Azure.Messaging.ServiceBus.Administration;

namespace TopicsAndSubscriptions.Helper
{
    1 reference | 0 changes | 0 authors, 0 changes
    public class QueueManager
    {
        ServiceBusAdministrationClient client;
        0 references | 0 changes | 0 authors, 0 changes
        public QueueManager(string connectionString)
        {
            client = new ServiceBusAdministrationClient(connectionString);

        }
        0 references | 0 changes | 0 authors, 0 changes
        public async Task<QueueProperties> CreateQueueAsync(string queueName)
        {
            if (await client.QueueExistsAsync(queueName) == true)
            {
                await client.DeleteQueueAsync(queueName);
            }
            var queueOptions = new CreateQueueOptions(queueName)
            {
                AutoDeleteOnIdle = TimeSpan.FromDays(7),
                DefaultMessageTimeToLive = TimeSpan.FromDays(2),
                DuplicateDetectionHistoryTimeWindow = TimeSpan.FromMinutes(1),
                EnableBatchedOperations = true,
                EnablePartitioning = false,
                MaxSizeInMegabytes = 2048,
                RequiresDuplicateDetection = true,
                UserMetadata = "some metadata"
            };

            queueOptions.AuthorizationRules.Add(new SharedAccessAuthorizationRule(
                   "allClaims",
                   new[] { AccessRights.Manage, AccessRights.Send, AccessRights.Listen }));

            var createdQueue = await client.CreateQueueAsync(queueOptions);
            return createdQueue;
        }
    }
}
```

Azure Service Bus – Duplicate Messages

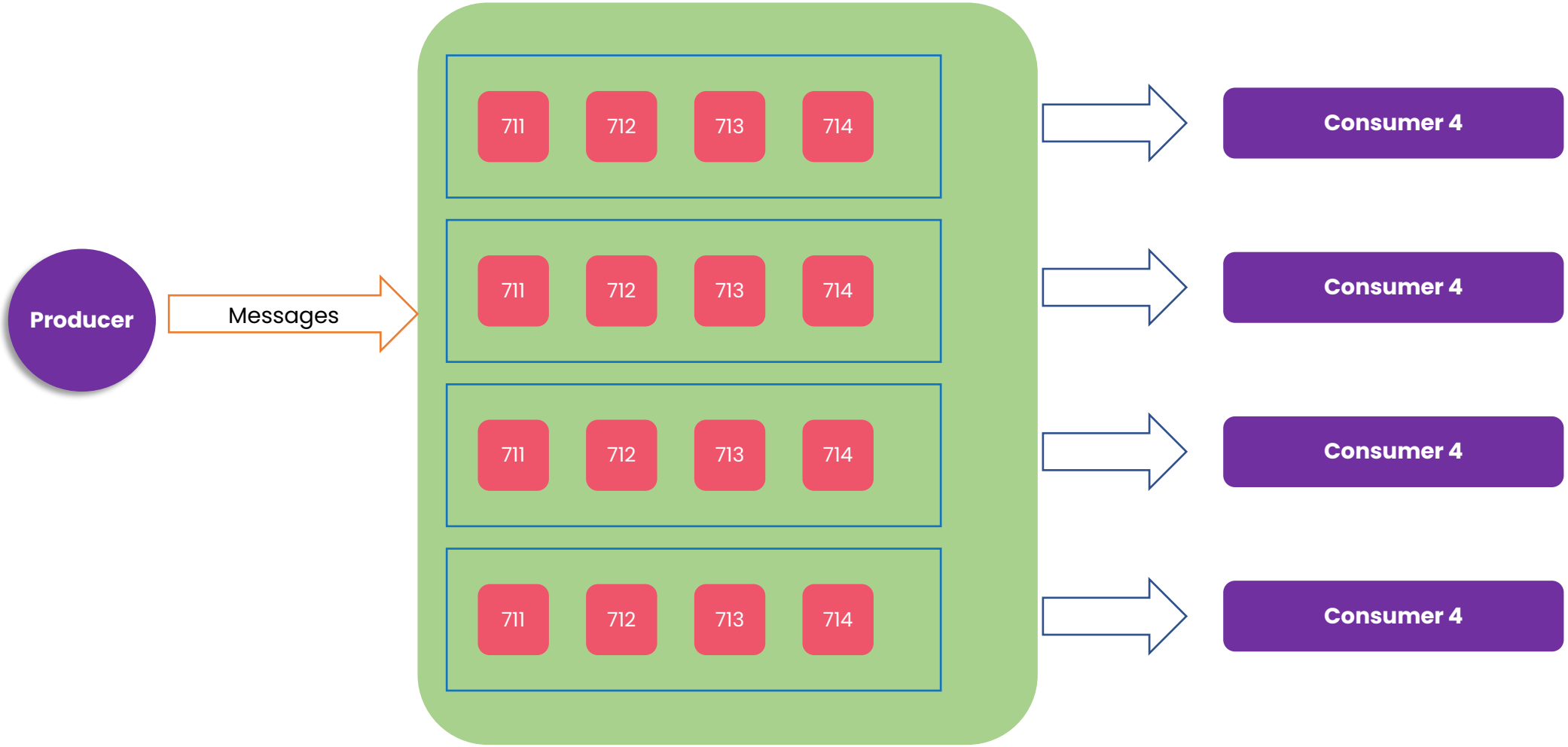
PowerShell

```
az servicebus queue create \
    --resource-group myresourcegroup \
    --namespace-name mynamespace \
    --name myqueue \
    --enable-duplicate-detection true \
    --duplicate-detection-history-time-window P1D
```

Azure Service Bus – Duplicate Messages

Demo

Azure Service Bus – Topic



Azure Service Bus – Topics – Filters/Actions

SQL Filters – User Defined Properties

Correlation Filters – System Defined properties

- ContentType
- Label
- MessageId
- ReplyTo
- ReplyToSessionId
- SessionId
- To

Actions

```
CreateRuleOptions createRuleOptions;
createRuleOptions = new CreateRuleOptions
{
    Name = item.StateCode + "-Rule",
    Filter = new SqlRuleFilter($"State='{item.StateCode}'")
};
//Increase the price by 20% if State =MH

if (item.StateCode == "MH")
{
    createRuleOptions.Action = new SqlRuleAction("SET TotalPrice=TotalPrice+TotalPrice*.20");
}
```

Azure Service Bus – Publish Subscribe

Demo

Service Bus authentication and authorization

Service Bus has two authorization models.

1. Azure Active Directory

- Roles can be assigned at namespace and entity level
- Authenticate with managed identities : https://learn.microsoft.com/en-us/azure/service-bus-messaging/service-bus-managed-service-identity
- Authenticate from an application : https://learn.microsoft.com/en-us/azure/service-bus-messaging/authenticate-application

2. Shared access signature

- Simple for External Clients, that can not use Azure Active Directory
- Rights management at namespace or queue or topic : send , Listen and Manage
- HMAC-SHA256 Signed Tokens are issued using name and key
- 12 named rules per scope : Send, Listen and Manage

Service Bus dead-letter queues

- Expired Message
- Undelivered messages
- Errors while processing subscription rules
- Maximum delivery count

Dead-letter reason	Dead-letter error description
HeaderSizeExceeded	The size quota for this stream has been exceeded.
TTLExpiredException	The message expired and was dead lettered. See the Time to live section for details.
Session ID is null	Session enabled entity doesn't allow a message whose session identifier is null.
MaxTransferHopCountExceeded	The maximum number of allowed hops when forwarding between queues has been exceeded. This value is set to 4.
MaxDeliveryCountExceededExceptionMessage	Message couldn't be consumed after maximum delivery attempts. See the Maximum delivery count section for details.

Service Bus dead-letter queues

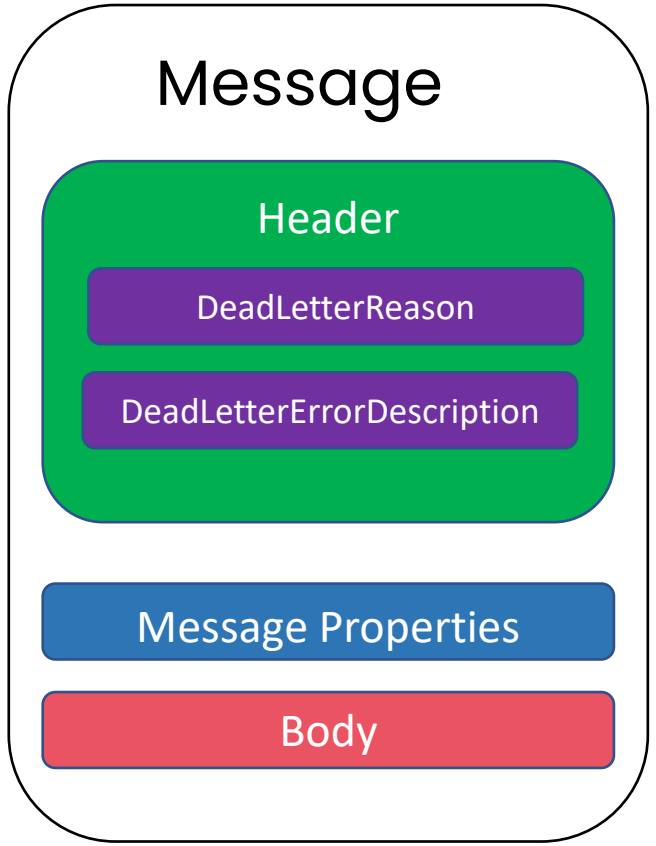
Application-level dead-lettering

- Poison Message
- Processing Failures

Path to the dead-letter queue

<queue path>/$deadletterqueue

<topic path>/Subscriptions/<subscription path>/$deadletterqueue




```
private readonly ServiceBusReceiver serviceBusReceiver;

serviceBusReceiver = ServiceBusClient.CreateReceiver(connectionString);

await serviceBusReceiver.DeadLetterMessageAsync(servicebusMessage,"Error");
```

Azure Service Bus Tiers

	Basic	Standard	Premium
Queues	Yes	Yes	Yes
Scheduled messages	Yes	Yes	Yes
Topics		Yes	Yes
Transactions		Yes	Yes
De-Duplication		Yes	Yes
Sessions		Yes	Yes
ForwardTo/SendVia		Yes	Yes
Message Size	256 KB	256 KB	256 KB
Resource Isolation			Yes
Geo-Disaster Recovery (Geo-DR)			Yes
Java Messaging Service (JMS) 2.0 Support			Yes
Availability Zones (AZ) support			Yes

https://azure.microsoft.com/en-in/pricing/details/service-bus/

Azure Service Bus Pricing

https://azure.microsoft.com/en-in/pricing/details/service-bus/

Basic	
Operations	**$0.05** per million operations
Standard	
Base charge [1]	**$0.0135**/hour
First 13M ops/month	Included
Next 87M ops (13-100M ops)/month	**$0.80** per million operations
Next 2,400M ops (100-2500M ops)/month	**$0.50** per million operations
Over 2,500M ops/month	**$0.20** per million operations
Premium	
Hourly	**$0.928**/hour

SDK Supported

.NET

JAVA

Spring

Node.js

Python

PHP

How to use Service Bus topics and subscriptions with PHP

Article • 09/21/2022 • 11 minutes to read • 12 contributors

This article shows you how to use Service Bus topics and subscriptions. The samples are written in PHP and use the Azure SDK for PHP. The scenarios covered include:

- Creating topics and subscriptions
- Creating subscription filters
- Sending messages to a topic
- Receiving messages from a subscription
- Deleting topics and subscriptions

The following video describes how to integrate Spring JMS applications with Azure Service Bus using JMS 2.0.

Performance and Reliability

Throughput:

Service Bus Standard has a soft throttle at about 500 msg/sec per namespace

- 1000 credits per second, each send and receive operation counts one credit.

Service Bus Premium is only limited by compute and memory (MU) as well at I/O caps

- One log (a single queue) can handle about 10 MB/sec data I/O combined (5000 msg/sec @ 1kB)
- More features, more CPU and memory use, less throughput.

Reliability:

- Monthly global uptime (are endpoints reachable?): **100%**
- Monthly global reliability (are operations succeeding?): **> 99.995%**

Service Bus Premium Messaging .NET Performance Test

https://github.com/Azure-Samples/service-bus-dotnet-messaging-performance

Service Bus Queues-Transaction

Q & A