

Deep Dive with Azure Service Bus

@danielmarbach

Events



Event Grid

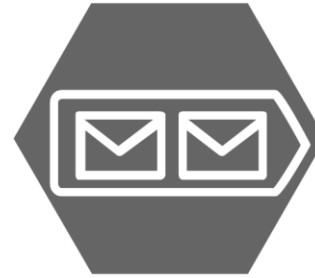
Cross cloud reactive
eventing



Event Hubs

Big data streaming

Messages



Storage Queues

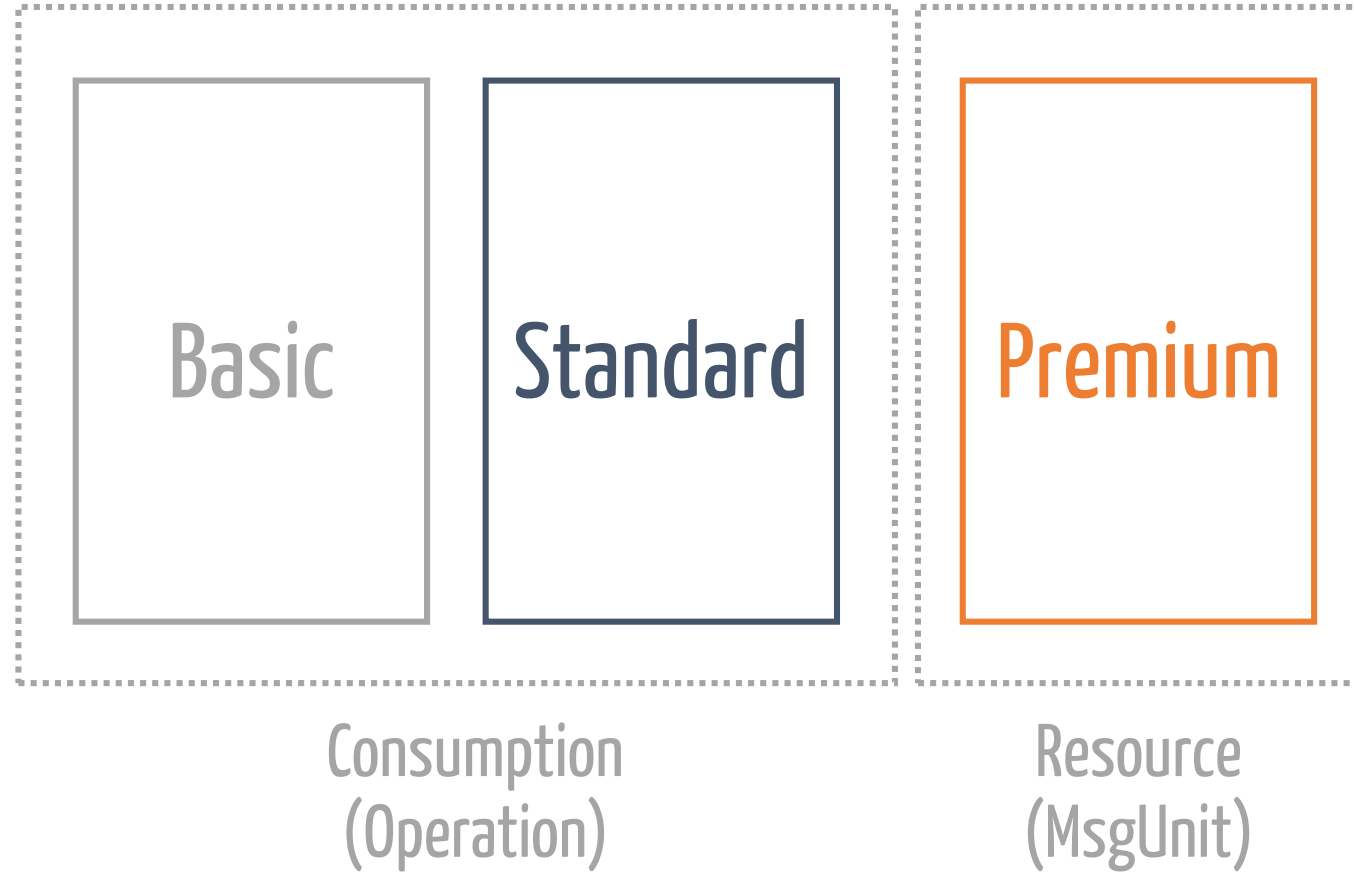
Simple task queues



Service Bus

Enterprise messaging

Tiers







2,905 packages returned for azure servicebus

☒ **Include prerelease**

WindowsAzure.ServiceBus by: [microsoft nugetservicebus](#)

Microsoft Azure Service Bus

↓ 12,158,973 total downloads | ⌚ last updated 2 months ago | 📦 Latest version: 5.0.0 | 🔗 [ServiceBus Microsoft Azure AppFabric Messaging PubSub P...](#)

Use this for Microsoft Azure Service Bus Queues, Topics, EventHub and Relay backend operations. It adds Microsoft.ServiceBus.dll along with related configuration files to your project. This library allows AMQP 1.0 to be used as one of the protocols for communication with Microsoft Azure... [More information](#)



WindowsAzure.Storage by: [azure-sdk microsoft](#)

Windows Azure Storage

↓ 27,305,814 total downloads | ⌚ last updated 16 days ago | 📦 Latest version: 9.3.1 | 🔗 [Microsoft Azure Storage Table Blob File Queue Scalable wind...](#)

This client library enables working with the Microsoft Azure storage services which include the blob and file service for storing binary and text data, the table service for storing structured non-relational data, and the queue service for storing messages that may be accessed by a client. For... [More information](#)



Microsoft.Azure.Jobs.ServiceBus by: [aspnet microsoft](#)

↓ 6,118 total downloads | ⌚ last updated 2014-08-21 | 📦 Latest version: 0.3.2-beta | 🔗 [Microsoft Azure WebJobs Jobs ServiceBus](#)

Legacy package, Microsoft.Azure.Jobs.ServiceBus is now included in the 'Microsoft.Azure.WebJobs.ServiceBus' package.



Microsoft.Azure.ServiceBus by: [microsoft azure-sdk nugetservicebus](#)

↓ 899,624 total downloads | ⌚ last updated 14 days ago | 📦 Latest version: 3.1.0 | 🔗 [Azure Service Bus ServiceBus .NET AMQP IoT Queue Topic](#)

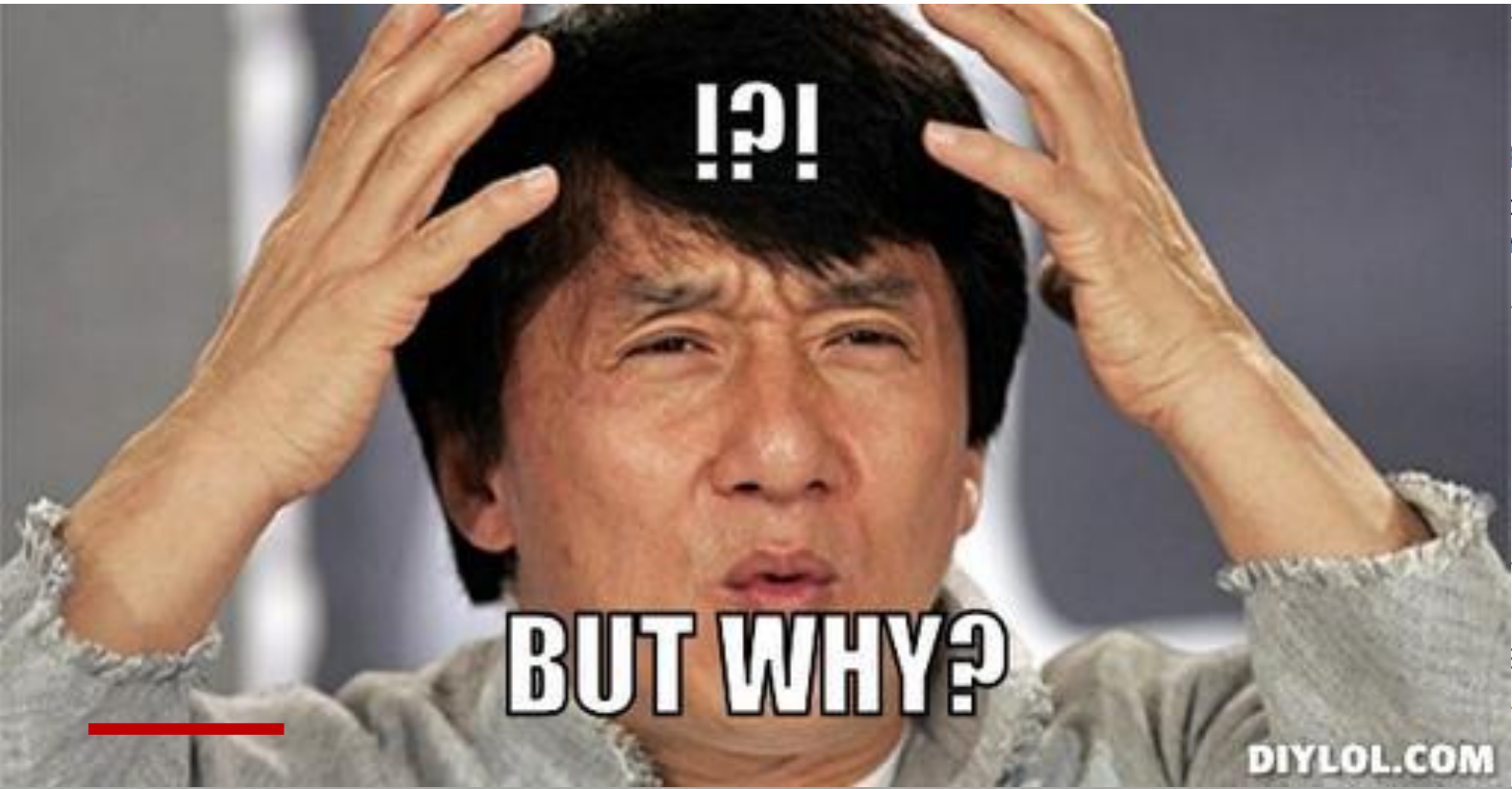
This is the next generation Azure Service Bus .NET Standard client library that focuses on queues & topics. For more information about Service Bus, see <https://azure.microsoft.com/en-us/services/service-bus/>

1

Windows
Microsoft
↓ 12,15
Use this
configu
inform

2

Microsoft
↓ 8
This
http



toFabric Messaging PubSub P...
dll along with related
rosoft Azure... [More](#)

ET AMQP IoT Queue Topic
tion about Service Bus, see

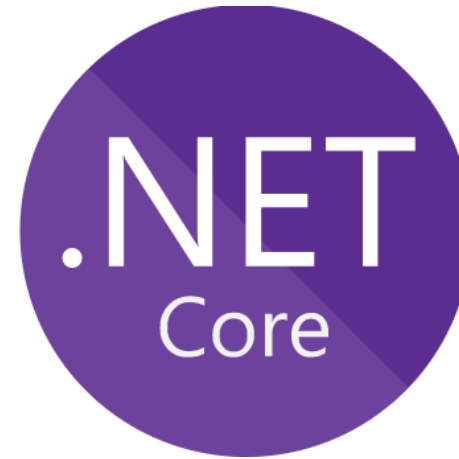
Microsoft ❤️ OpenSource

Microsoft.Azure.ServiceBus

.Net Standard



WindowsAzure.ServiceBus



azure servicebus



3,600 packages returned for azure servicebus

☒ Include prereleaseMicrosoft.Azure.Jobs.ServiceBus  by: [aspnet](#) [Microsoft](#)

↓ 8,600 total downloads | ⌚ last updated 8/21/2014 | 📦 Latest version: 0.3.2-beta | 🔗 Microsoft Azure WebJobs Jobs ServiceBus
Legacy package, Microsoft.Azure.Jobs.ServiceBus is now included in the 'Microsoft.Azure.WebJobs.ServiceBus' package.

Microsoft.Azure.ServiceBus  by: [azure-sdk](#) [nugetservicebus](#) [Microsoft](#)

↓ 4,314,906 total downloads | ⌚ last updated 2 months ago | 📦 Latest version: 3.4.0 | 🔗 Azure Service Bus ServiceBus .NET AMQP IoT Queue Topic
This is the next generation Azure Service Bus .NET Standard client library that focuses on queues & topics. For more information about Service Bus, see <https://azure.microsoft.com/en-us/services/service-bus/>

Microsoft.Azure.Management.ServiceBus  by: [azure-sdk](#) [Microsoft](#)

↓ 596,196 total downloads | ⌚ last updated 2 months ago | 📦 Latest version: 2.1.0 | 🔗 Microsoft Azure ServiceBus Management management SHA25...
Provides developers with libraries to create and manage Namespaces and manage Authorization Rules. Note: This client library is for ServiceBus under Azure Resource Manager.



github.com/azure/azure-sdk-for-net

AMQP 1.0

Send

```
<PackageReference Include="Microsoft.Azure.ServiceBus" Version="3.4.0" />
```













```
var client = new QueueClient(connectionString, destination);  
var message = new Message();  
message.Body = Encoding.UTF8.GetBytes("Deep Dive");  
  
message.UserProperties.Add("TenantId", "MyTenantId");  
  
await client.SendAsync(message);  
Console.WriteLine("Sent message");  
await client.CloseAsync();
```

```
message.UserProperties.Add("TenantId", "MyTenantId");
```

```
message.
```

```
await cl
```

```
Console.
```

 Body	by
 Clone	
 ContentType	Ge
 CorrelationId	
 Equals	Re
 ExpiresAtUtc	Th
 GetHashCode	str
 GetType	
 Label	
 MessageId	Sy:
 PartitionKey	
 ReplyTo	

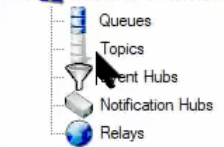
```
c:\p\AzureServiceBus.DeepDive\Send
```

```
> dotnet run
```



Service Bus Namespace

sb://nservicebustests.servicebus.windows.net/



View Queue: queue

Log

<20:37:38> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.
<20:37:38> MessagingFactory successfully created

```
c:\p\AzureServiceBus.DeepDive\Send
```

```
> dotnet run
```



Service Bus Namespace

sb://nservicebustests.servicebus.windows.net/
Queues
queue (1, 0, 0)
Topics
Event Hubs
Notification Hubs
Relays

View Queue: queue

Description

Authorization Rules

Path

Relative URI:
queue

Auto Delete On Idle

Days: 106751
Hours: 2
Minutes: 48
Seconds: 5
Milliseconds: 477

Duplicate Detection History Time Window

Days: 0
Hours: 0
Minutes: 10
Seconds: 0
Milliseconds: 0

Default Message Time To Live

Days: 106751
Hours: 2
Minutes: 48
Seconds: 5
Milliseconds: 477

Queue Properties

Max Queue Size In GB:
1 GB

Max Delivery Count:
10

User Description:
...

Forward To:
...

Forward Dead Lettered Messages To:
...

Lock Duration

Days: 0
Hours: 0
Minutes: 1
Seconds: 0
Milliseconds: 0

Queue Settings

☐ Enable Batched Operations
☐ Enable Dead Lettering On Message Expiration
☐ Enable Partitioning
☐ Enable Express
☐ Requires Duplicate Detection
☐ Requires Session
☐ Enforce Message Ordering

Queue Information

Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	201
Created At	14.06.2019 18:38:48
Accessed At	14.06.2019 18:40:10
Updated At	14.06.2019 18:38:48
Active Message Count	1
DeadLetter Message Count	0
Scheduled Message Count	0
Transfer Message Count	0
Transfer DL Message Count	0
Message Count	1

Purge

Purge DLQ

Messages

Deadletter

Transf DLQ

Refresh

Status

Delete

Update

Log

```
<20:37:38> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.  
<20:37:38> MessagingFactory successfully created  
<20:38:51> The queue queue has been successfully created.  
<20:39:51> The queue queue has been successfully retrieved.  
<20:40:03> The queue queue has been successfully retrieved.  
<20:40:03> [2] messages have been purged from the [queue] queue in [1602] milliseconds.  
<20:40:20> The queue queue has been successfully retrieved.
```

```
<PackageReference Include="Microsoft.Azure.Management.ServiceBus" Version="2.1.0" />
```

```
var client = new ManagementClient(connectionString);  
await client.CreateQueueAsync(destination);  
await client.CloseAsync();
```

Receive

```
var client = new QueueClient(connectionString, destination);  
await client.SendAsync(new Message(Encoding.UTF8.GetBytes("Deep Dive")));  
Console.WriteLine("Message sent");
```

```
client.RegisterMessageHandler(  
    async (message, token) =>  
    {  
        Console.WriteLine(  
            $"Received message with '{message.MessageId}' and content '{Encoding.UTF8.GetString(message.Body)}'");  
        // throw new InvalidOperationException();  
        await client.CompleteAsync(message.SystemProperties.LockToken);  
    },  
    new MessageHandlerOptions(  
        exception =>  
        {  
            Console.WriteLine($"Exception: {exception.Exception}");  
            Console.WriteLine($"Action: {exception.ExceptionReceivedContext.Action}");  
            Console.WriteLine($"ClientId: {exception.ExceptionReceivedContext.ClientId}");  
            Console.WriteLine($"Endpoint: {exception.ExceptionReceivedContext.Endpoint}");  
            Console.WriteLine($"EntityPath: {exception.ExceptionReceivedContext.EntityPath}");  
            return Task.CompletedTask;  
        })  
    {  
        AutoComplete = false,  
        MaxConcurrentCalls = 1,  
        MaxAutoRenewDuration = TimeSpan.FromMinutes(10)  
    }  
);
```

```
c:\p\AzureServiceBus.DeepDive\Receive
```

```
> dotnet run
```



```
c:\p\AzureServiceBus.DeepDive\Receive
```

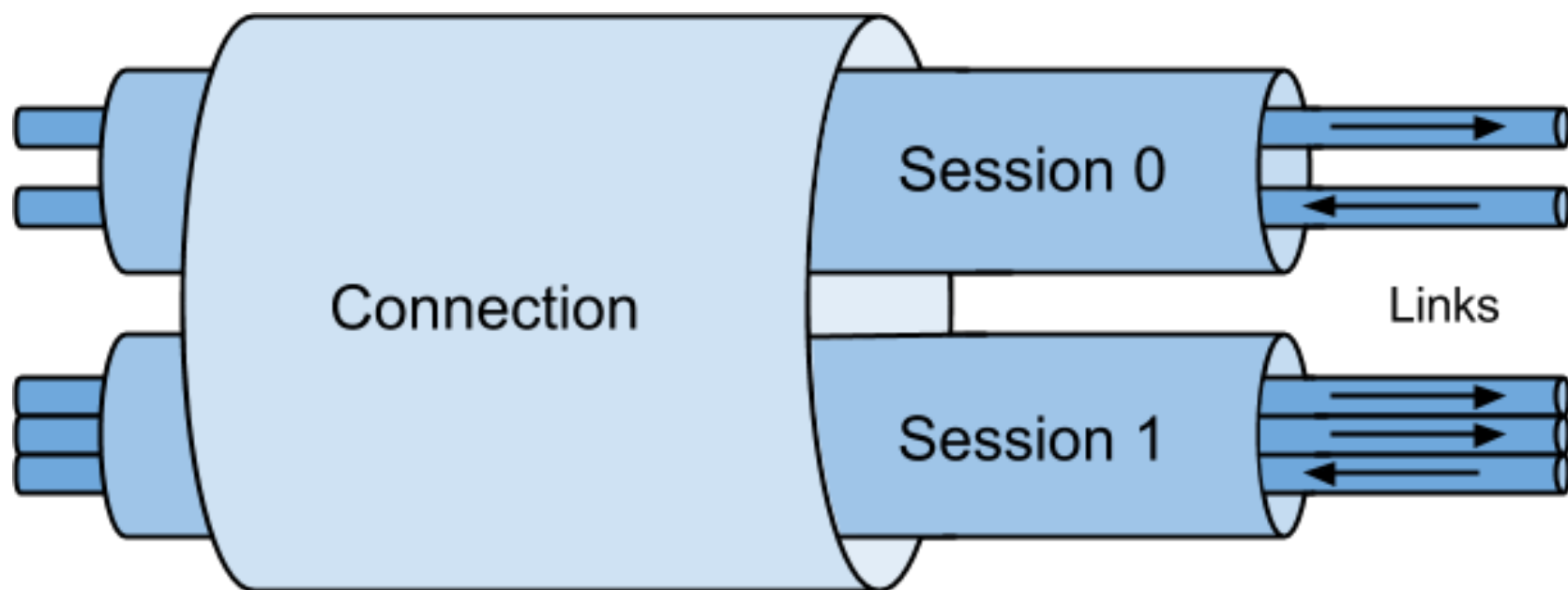
```
> dotnet run
```

```
|
```



MessageSender
MessageReceiver

QueueClient
TopicClient
SubscriptionClient



Connections

```
var sender = new MessageSender(connectionString, destination);  
await sender.SendAsync(new Message(Encoding.UTF8.GetBytes("Deep Dive")));  
var receiver = new MessageReceiver(connectionString, destination);  
await receiver.ReceiveAsync();
```

```
var connection = new ServiceBusConnection(connectionString);  
sender = new MessageSender(connection, destination);  
receiver = new MessageReceiver(connection, destination);  
  
await sender.SendAsync(new Message(Encoding.UTF8.GetBytes("Deep Dive")));  
await receiver.ReceiveAsync();
```

```
c:\p\AzureServiceBus.DeepDive\Connections
```

```
> dotnet run
```



C:\Users\Daniel

> netstat -na | find "5671"|



```
c:\p\AzureServiceBus.DeepDive\Connections
```

```
> dotnet run
```

```
netstat -na | find "5671"
```

```
Continue with connection sharing
```

```
|
```



C:\Users\Daniel

> netstat -na | find "5671"

TCP	192.168.1.14:52249	52.166.127.37:5671	ESTABLISHED
TCP	192.168.1.14:52250	52.166.127.37:5671	ESTABLISHED

C:\Users\Daniel

> netstat -na | find "5671"|

Scheduling

```
var sender = new MessageSender(connectionString, destination);  
var due = DateTimeOffset.UtcNow.AddSeconds(10);  
await sender.ScheduleMessageAsync(new Message(Encoding.UTF8.GetBytes($"Deep Dive + {due}")), due);  
Console.WriteLine($"{DateTimeOffset.UtcNow}: Message scheduled first");
```

```
var sequenceId =  
|   await sender.ScheduleMessageAsync(new Message(Encoding.UTF8.GetBytes($"Deep Dive + {due}")), due);  
Console.WriteLine($"{DateTimeOffset.UtcNow}: Message scheduled second");  
  
await sender.CancelScheduledMessageAsync(sequenceId);  
Console.WriteLine($"{DateTimeOffset.UtcNow}: Canceled second");
```

```
c:\p\AzureServiceBus.DeepDive\Scheduling  
> dotnet run
```



Expiry

```
var client = new QueueClient(connectionString, destination);

var message = new Message();
message.Body = Encoding.UTF8.GetBytes("Half life");
// if not set the default time to live on the queue counts
message.TimeToLive = TimeSpan.FromSeconds(10);

await client.SendAsync(message);

// Note that expired messages are only purged and moved to the DLQ when there is at least one
// active receiver pulling from the main queue or subscription; that behavior is by design.
await Prepare.SimulateActiveReceiver(client);
```

```
c:\p\AzureServiceBus.DeepDive\Expiry  
> dotnet run
```



Where does the message go?

Service Bus Namespace

sb://servicebustests.servicebus.windows.net/

Queues

queue (0/0)

TopicsEvent HubsNotification HubsRelays

View Queue: queue

Description

Authorization Rules

Path

Relative URI:

queue

Auto Delete On Idle

Days:106751

Hours:2

Minutes:48

Seconds:5

Milliseconds:477

Duplicate Detection History Time Window

Days:0

Hours:0

Minutes:10

Seconds:0

Milliseconds:0

Default Message Time To Live

Days:106751

Hours:2

Minutes:48

Seconds:5

Milliseconds:477

Queue Properties

Max Queue Size In GB:

1 GB

Max Delivery Count:

2147483647

User Description:

Forward To:

Forward Dead Lettered Messages To:

Lock Duration

Days:0

Hours:0

Minutes:1

Seconds:0

Milliseconds:0

Queue Settings

☒ Enable Batched Operations

☐ Enable Dead Lettering On Message Expiration

☐ Enable Partitioning

☐ Enable Express

☐ Requires Duplicate Detection

☐ Requires Session

☒ Enforce Message Ordering

Queue Information

Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	0
Created At	14.06.2019 20:41:43
Accessed At	14.06.2019 20:41:59
Updated At	14.06.2019 20:41:43
Active Message Count	0
DeadLetter Message Count	0
Scheduled Message Count	0
Transfer Message Count	0
Transfer DL Message Count	0
Message Count	0

Purge

Purge DLQ

Messages

Deadletter

Transf DLQ

Refresh

Status

Delete

Update

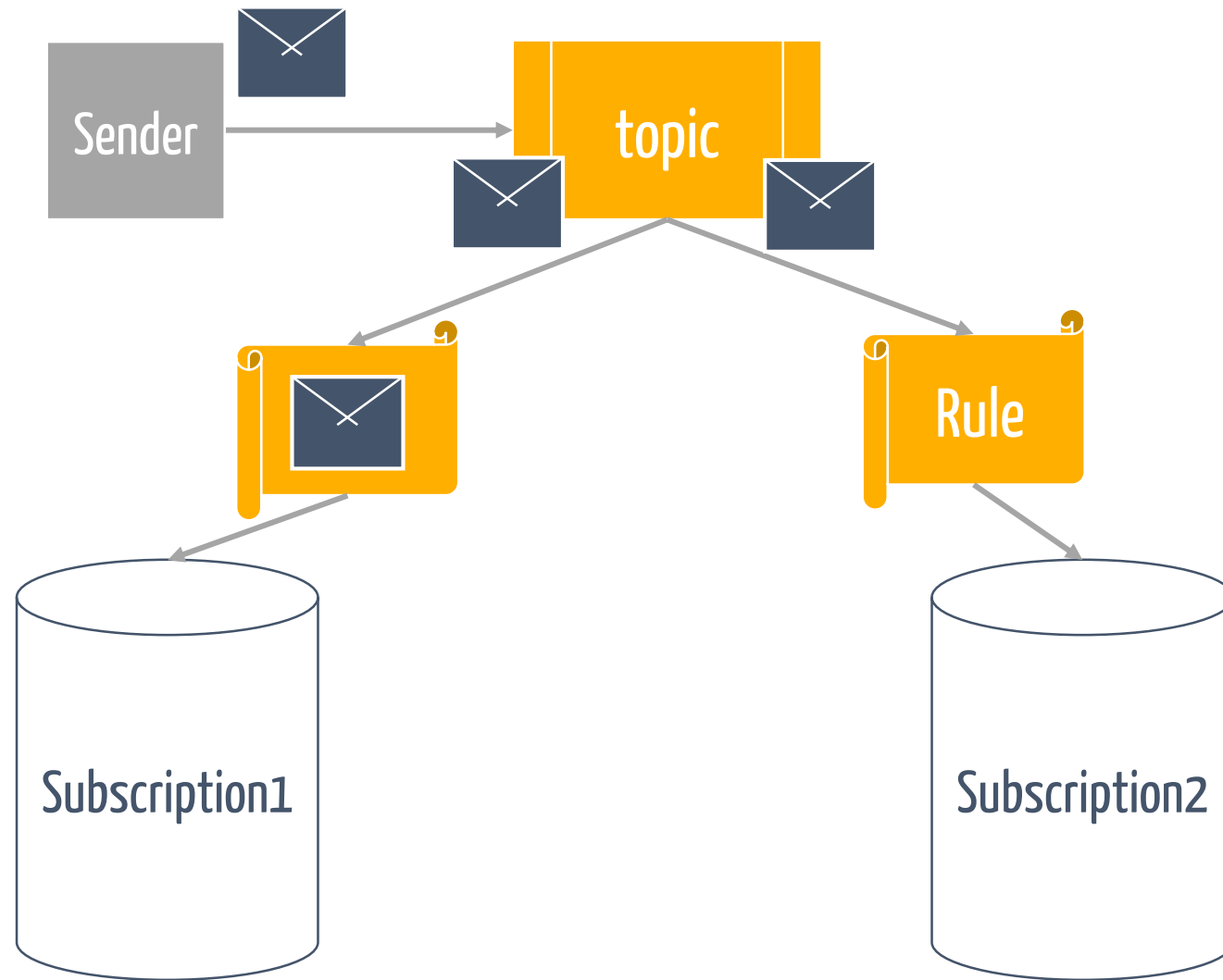
Log

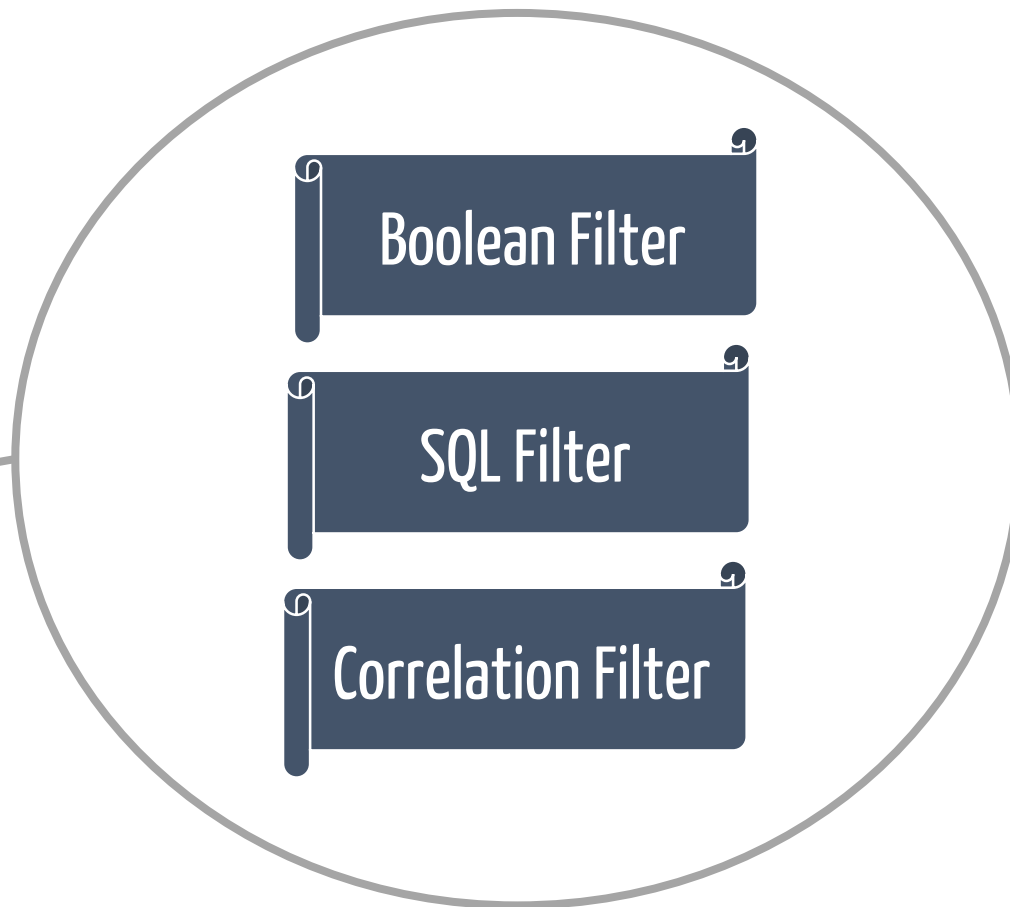
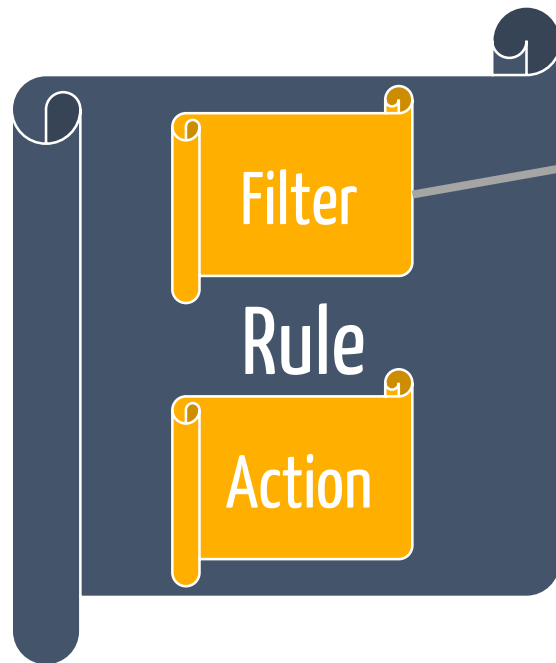
<21:02:00> The queue queue has been successfully deleted.
<22:42:14> The queue queue has been successfully retrieved.
<22:42:15> The queue queue has been successfully retrieved.
<22:42:16> The queue queue has been successfully retrieved.
<22:42:16> The queue queue has been successfully retrieved.
<22:42:16> The queue queue has been successfully retrieved.
<22:42:16> The queue queue has been successfully retrieved.
<22:42:16> The queue queue has been successfully retrieved.
<22:42:17> The queue queue has been successfully retrieved.
<22:42:17> The queue queue has been successfully retrieved.
<22:42:18> The queue queue has been successfully retrieved.
<22:42:18> The queue queue has been successfully retrieved.
<22:42:18> The queue queue has been successfully retrieved.
<22:42:18> The queue queue has been successfully retrieved.
<22:42:19> The queue queue has been successfully retrieved.
<22:42:19> The queue queue has been successfully retrieved.
<22:42:19> The queue queue has been successfully retrieved.
<22:42:19> The queue queue has been successfully retrieved.

Deadlettering

Forwarding

Pub/Sub

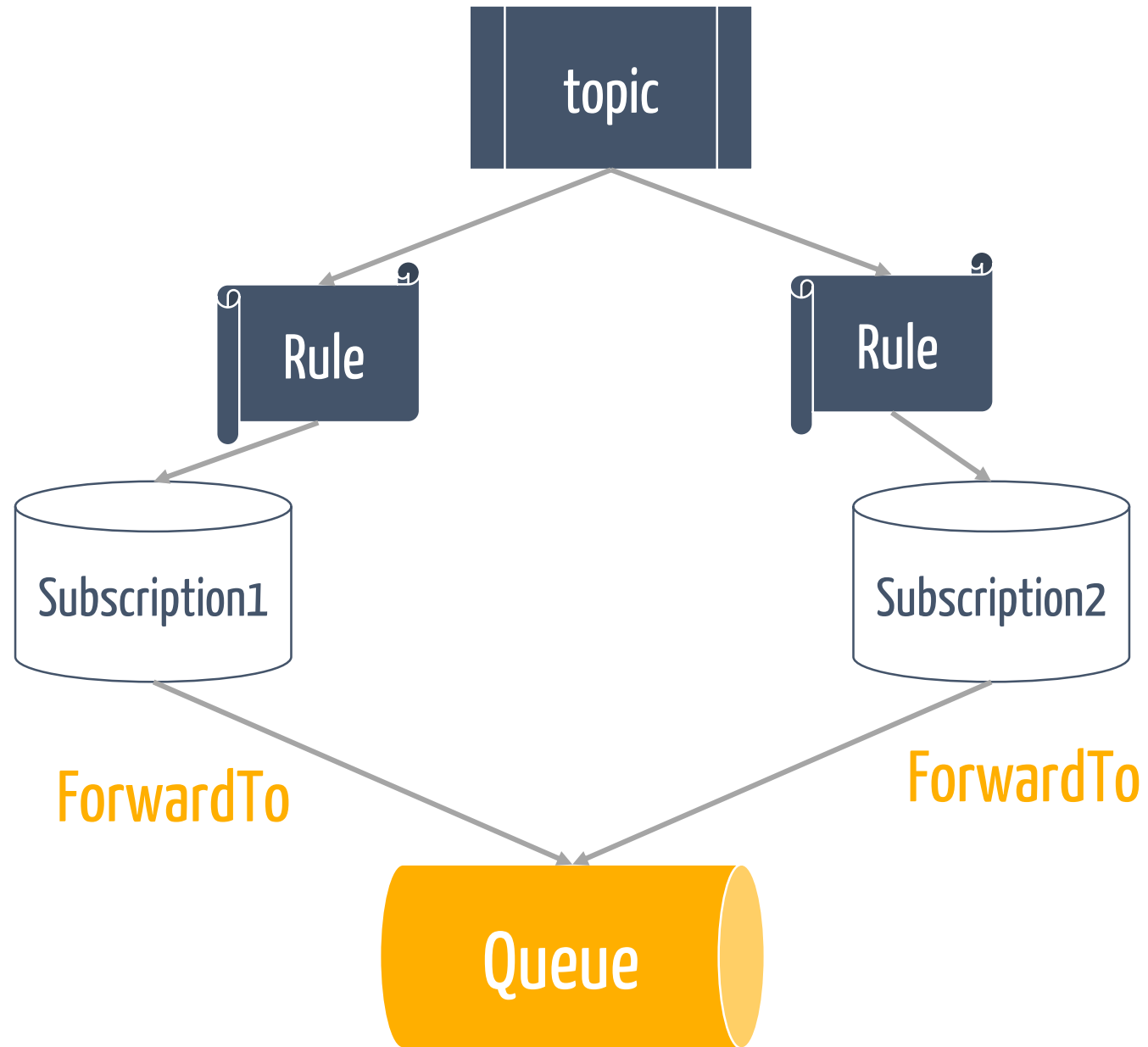




Favor **Correlation filter** over SQL filter

Subscriptions are **virtual queues** and
subscribers need to **receive** from them

Topologies



Atomic Sends

Batching

Upgrade to premium tier

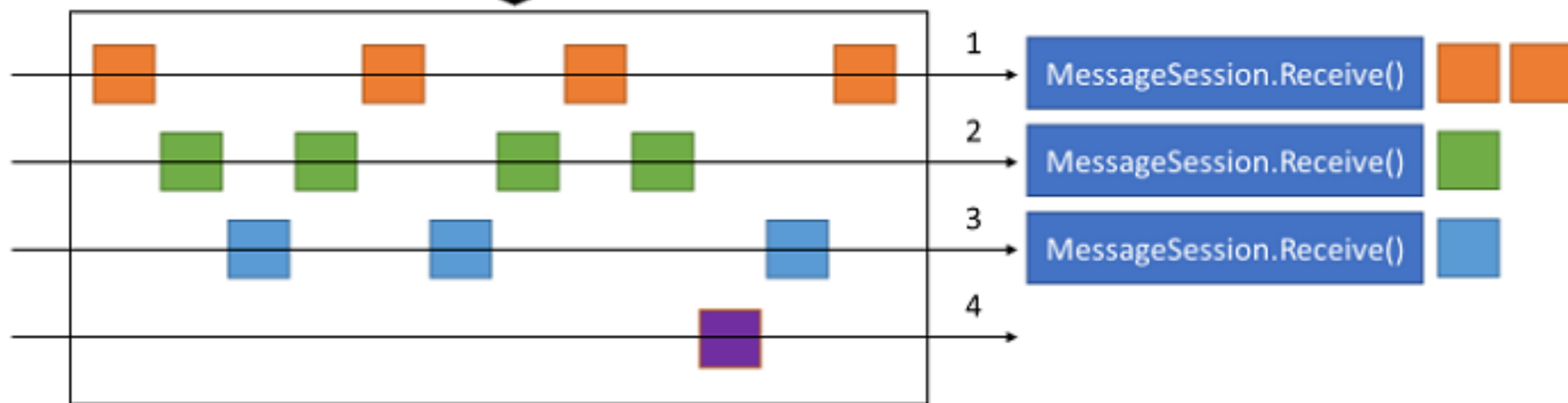
Use `ServiceBus.AttachmentPlugin`

Send Via



TransferDLQ

Sessions



Dedup

Plugins

Thanks

Slides, Links...

github.com/danielmarbach/AzureServiceBus.DeepDive

Q & A



Software Engineer
Microsoft MVP

@danielmarbach
particular.net/blog
planetgeek.ch

Thanks