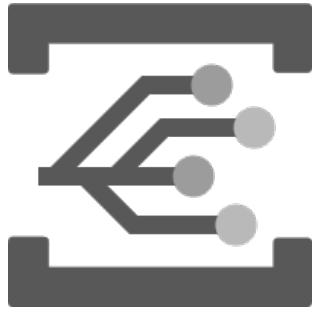




Deep Dive with Azure Service Bus

@danielmarbach

Eventing



Event Grid

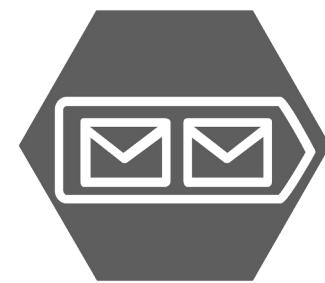
Cross cloud reactive
eventing



Event Hubs

Big data streaming

Messaging



Storage Queues

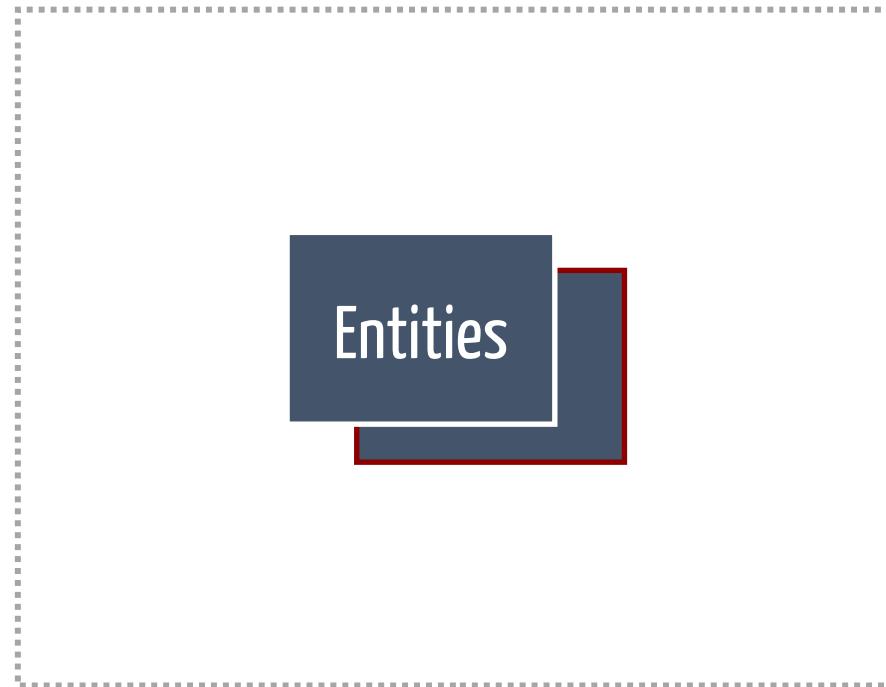
Simple task queues



Service Bus

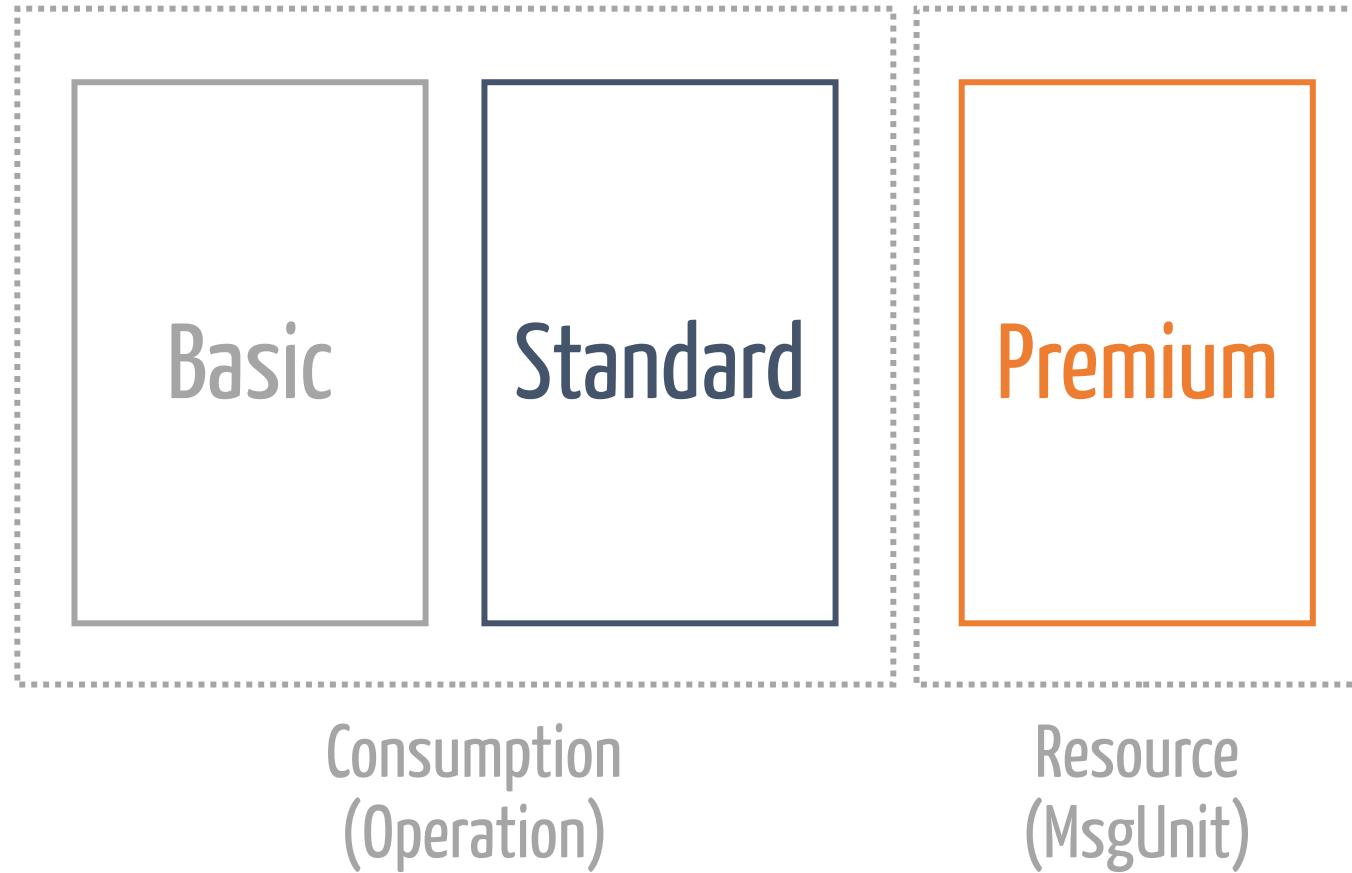
Enterprise messaging

Namespace



Unit of Isolation

Tiers



azure service bus



468 packages returned for azure service bus

[Filter](#)

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

↓ 24,600,513 total downloads last updated a month ago Latest version: 7.11.0-beta.1

[Azure Service Bus](#) [ServiceBus](#) [.NET](#) [AMQP](#) [windows](#) [azureofficial](#) [azureofficial](#)

Azure Service Bus is a fully managed enterprise integration message broker. Service Bus can decouple applications and services. Service Bus offers a reliable and secure platform for asynchronous transfer of data and state. This client library allows for both sending and receiving messages using... [More information](#)



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

↓ 9,457 total downloads last updated 3 months ago Latest version: 1.0.0

[azure management](#) [arm](#) [resource manager](#) [azure.resourcemanager.servicebus](#)

Azure management client SDK for Azure resource provider Microsoft.ServiceBus. This is a stable version. This version uses a next-generation code generator that introduces important breaking changes, but also new features (such as intuitive authentication, custom HTTP pipeline, distributed... [More information](#)



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

↓ 17,731,598 total downloads last updated 2 months ago Latest version: 5.7.0

Microsoft Azure WebJobs SDK ServiceBus Extension



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

↓ 7,856,539 total downloads last updated 6/8/2021 Latest version: 5.0.1

[ServiceBus](#) [Microsoft](#) [Azure](#) [AppFabric](#) [Messaging](#) [PubSub](#) [Publish](#) [Subscribe](#) [Queue](#) [Topic](#) [More tags](#)

Use this for EventHub distributed partition processing. For more information please visit <http://go.microsoft.com/fwlink/?LinkID=403957> Please note that a newer package Azure.Messaging.EventHubs.Processor is available as of February 2020. While this package will continue to receive critical... [More information](#)

Send

```
await using var serviceBusClient = new ServiceBusClient(connectionString);
await using var client = serviceBusClient.CreateSender(destination);

var message = new ServiceBusMessage("Deep Dive");
<PackageReference Include="Azure.Messaging.ServiceBus" />
message.ApplicationProperties.Add("TenantId", "MyTenantId");

await client.SendMessageAsync(message);
```

message.

```
await cl.Console.  
    GetType  
    MessageId  
    PartitionKey  
    ReplyTo  
    ReplyToSessionId  
    ScheduledEnqueueTime  
    SessionId  
    Subject  
    TimeToLive  
    To  
    ToString  
    TransactionPartitionKey
```

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

```
Unhandled Exception: Microsoft.Azure.ServiceBus.MessagingEntityNotFoundException: Put token failed. status-code: 404, status-d  
The messaging entity 'sb://nservicebustests.servicebus.windows.net/queue' could not be found. TrackingId:3884f2d2-5841-4a05-8  
cd570_G5, SystemTracker:nservicebustests.servicebus.windows.net:queue, Timestamp:2019-06-14T18:38:14.  
   at Microsoft.Azure.ServiceBus.Core.MessageSender.OnSendAsync(IList`1 messageList) in C:\source\azure-service-bus-dotnet\src  
Azure.ServiceBus\Core\MessageSender.cs:line 567  
   at Microsoft.Azure.ServiceBus.RetryPolicy.RunOperation(Func`1 operation, TimeSpan operationTimeout) in C:\source\azure-serv  
net\src\Microsoft.Azure.ServiceBus\RetryPolicy.cs:line 85  
   at Microsoft.Azure.ServiceBus.RetryPolicy.RunOperation(Func`1 operation, TimeSpan operationTimeout) in C:\source\azure-serv  
net\src\Microsoft.Azure.ServiceBus\RetryPolicy.cs:line 107  
   at Microsoft.Azure.ServiceBus.Core.MessageSender.SendAsync(IList`1 messageList) in C:\source\azure-service-bus-dotnet\src\M  
ure.ServiceBus\Core\MessageSender.cs:line 257  
   at Send.Program.Main(String[] args) in c:\p\AzureServiceBus.DeepDive\Send\Program.cs:line 28  
   at Send.Program.Main(String[] args) in c:\p\AzureServiceBus.DeepDive\Send\Program.cs:line 33  
   at Send.Program.<Main>(String[] args)
```

Service Bus Explorer

File Edit Actions View Help

Service Bus Namespace

sb://nservicebustests.servicebus.windows.net/
Queues
Topics
Event Hubs
Notification Hubs
Relays

Create Queue

Description | Authorization Rules

Path

Relative URL:

Duplicate Detection History Time Window

Days: Hours: Minutes: Seconds: Millisecs:

Queue Properties

Max Queue Size In GB:

 1 GB

Max Delivery Count:

User Description:

 ...

Forward To:

 ...

Forward Dead Lettered Messages To:

 ...

Auto Delete On Idle

Days: Hours: Minutes: Seconds: Millisecs:

Default Message Time To Live

Days: Hours: Minutes: Seconds: Millisecs:

Lock Duration

Days: Hours: Minutes: Seconds: Millisecs:

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

Create

Cancel

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 Millisecs: 477

Duplicate Detection History Time Window

Days: 0 Hours: 0 Minutes: 10 Seconds: 0 Millisecs: 0

Default Message Time To Live

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

Queue Properties

Max Queue Size In GB: 1 GB

Max Delivery Count: 10

User Description:

Forward To:

Forward Dead Lettered Message:

Lock Duration

Days: 0 Hours: 0 Minutes: 1 Seconds: 0 Millisecs: 0

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

Retrieve messages from queue

Receive Mode

Peek Receive and Delete

Message Count

All Top 10

Message Inspector

Select a BrokeredMessage inspector...

Ok Cancel

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.
```

Management

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

Receive

```
await using var sender = serviceBusClient.CreateSender(destination);  
await sender.SendMessageAsync(new ServiceBusMessage("Deep Dive"));
```

```
var receiverOptions = new ServiceBusReceiverOptions
{
    ReceiveMode = ServiceBusReceiveMode.PeekLock,
    PrefetchCount = 10,
    SubQueue = SubQueue.None
};

await using var receiver = serviceBusClient
    .CreateReceiver(destination, receiverOptions);

await foreach (var message in receiver.ReceiveMessagesAsync())
{
    // consume message
}
```

```
var processorOptions = new ServiceBusProcessorOptions
{
    AutoCompleteMessages = false,
    MaxConcurrentCalls = 1,
    MaxAutoLockRenewalDuration = TimeSpan.FromMinutes(10),
    ReceiveMode = ServiceBusReceiveMode.PeekLock,
    PrefetchCount = 10
};

await using var receiver = serviceBusClient.
    CreateProcessor(destination, processorOptions);
```

```
receiver.ProcessMessageAsync += async messageEventArgs =>
{
    var message = messageEventArgs.Message;

    await Out.WriteLineAsync(
        $"Received message with '{message.MessageId}'  

        and content '{UTF8.GetString(message.Body)}'");

    // throw new InvalidOperationException();

    await messageEventArgs.CompleteMessageAsync(message);
};
```

```
receiver.ProcessErrorAsync += async errorEventArgs =>
{
    await Out.WriteLineAsync($"Exception:
        {errorEventArgs.Exception}"));
    await Out.WriteLineAsync($"FullyQualifiedNamespace:
{errorEventArgs.FullyQualifiedNamespace}"));
    await Out.WriteLineAsync($"ErrorSource:
        {errorEventArgs.ErrorSource}"));
    await Out.WriteLineAsync($"EntityPath:
        {errorEventArgs.EntityPath}"));
};
```

```
await receiver.StartProcessingAsync();
```

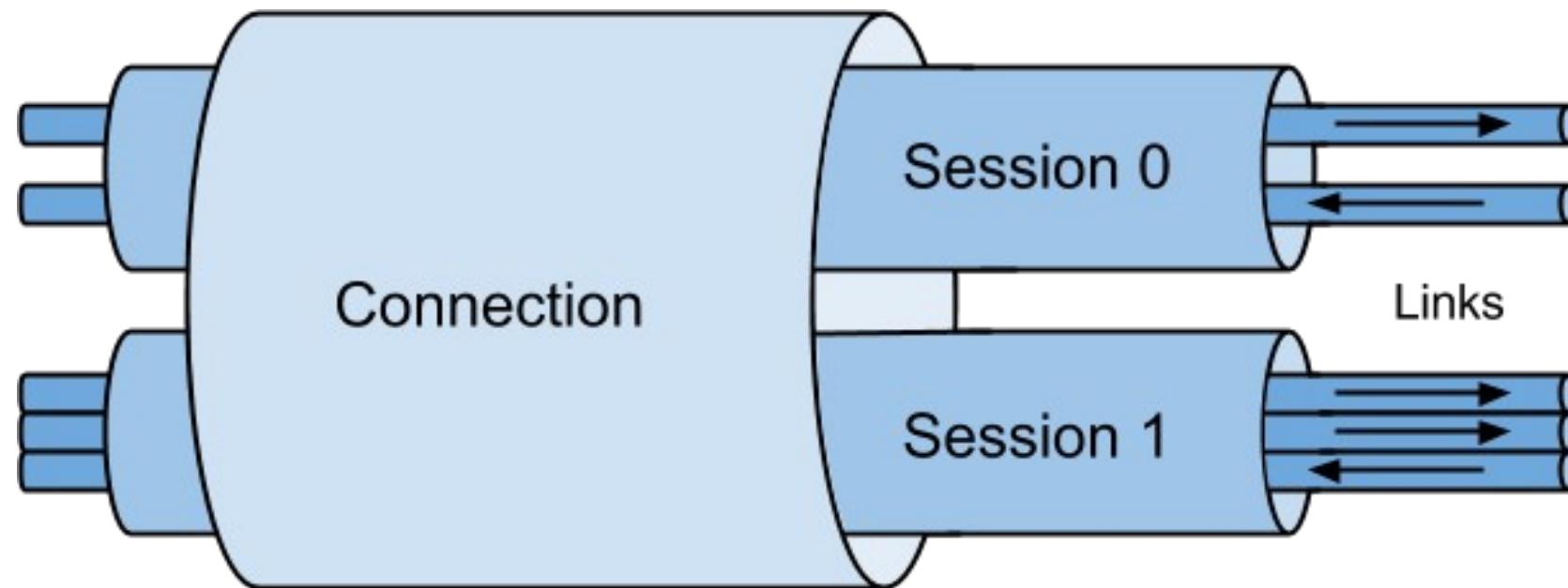
```
await untilStopRequired;
```

```
await receiver.StopProcessingAsync();
```

```
c:\p\AzureServiceBus.DeepDive\Receive
> dotnet run
Message sent
Received message with 'fe63aaa948444a08aa3c3c13b413c8cc' and content 'Deep Dive'
```

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

Connections



```
await using var serviceBusClient =
    new ServiceBusClient(connectionString);

await using var connectionSharingSender =
    serviceBusClient.CreateSender(destination);
await connectionSharingSender
    .SendMessageAsync(new ServiceBusMessage("Deep Dive"));

await using var connectionSharingReceiver =
    serviceBusClient.CreateReceiver(destination);

await connectionSharingReceiver.ReceiveMessageAsync();
```

```
C:\Users\Daniel  
> netstat -na | find "5671"  
TCP    192.168.1.14:52253      52.166.127.37:5671      ESTABLISHED
```

```
await using var senderServiceBusClient =
    new ServiceBusClient(connectionString);
await using var receiverServiceBusClient =
    new ServiceBusClient(connectionString);

await using var senderWithDedicatedConnection =
    senderServiceBusClient.CreateSender(destination);
await using var receiverWithDedicatedConnection =
    receiverServiceBusClient.CreateReceiver(destination);

await senderWithDedicatedConnection
    .SendMessageAsync(new ServiceBusMessage("Deep Dive"));
await receiverWithDedicatedConnection.ReceiveMessageAsync();
```

```
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
```

Scheduling

```
await using var sender = serviceBusClient.CreateSender(destination);

var due = DateTimeOffset.UtcNow.AddSeconds(10);
await sender.ScheduleMessageAsync(
    new ServiceBusMessage($"Deep Dive + {due}"), due);
```

```
var sequenceId = await sender.ScheduleMessageAsync(  
    new ServiceBusMessage($"Deep Dive + {due}"), due);  
  
await sender.CancelScheduledMessageAsync(sequenceId);
```

```
c:\p\AzureServiceBus.DeepDive\Scheduling
> dotnet run
14.06.2019 20:10:02 +00:00: Message scheduled first
14.06.2019 20:10:02 +00:00: Message scheduled second
14.06.2019 20:10:02 +00:00: Canceled second
```

Expiry

```
await using var sender = serviceBusClient.CreateSender(destination);

var message = new ServiceBusMessage("Half life")
{
    TimeToLive = TimeSpan.FromSeconds(10)
};

await sender.SendMessageAsync(message);

await Prepare.SimulateActiveReceiver(serviceBusClient, destination);
```

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

Where does the message go?

Service Bus Namespace

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

- Queues
- 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 Millisecs: 477

Duplicate Detection History Time Window

Days: 0 Hours: 0 Minutes: 10 Seconds: 0 Millisecs: 0

Default Message Time To Live

Days: 106751 Hours: 2 Minutes: 48 Seconds: 5 Millisecs: 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 Millisecs: 0

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

Queue Settings

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

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.
<22:42:17> 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.
<22:42:19> 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

```
var description = new CreateQueueOptions(destination)
{
    DeadLetteringOnMessageExpiration = true,
    MaxDeliveryCount = 1
};
await client.CreateQueueAsync(description);
```

```
var message = new ServiceBusMessage("Half life")
{
    TimeToLive = TimeSpan.FromSeconds(1)
};
await sender.SendMessageAsync(message);
```

```
var message = new ServiceBusMessage("Delivery Count");
await sender.SendMessageAsync(message);
```

```
var message = new ServiceBusMessage("Poor Soul");
message.ApplicationProperties.Add("Yehaa", "Why so happy?");
await sender.SendMessageAsync(message);
```

```
receiver.ProcessMessageAsync += async processMessageEventArgs =>
{
    var message = processMessageEventArgs.Message;
    switch (UTF8.GetString(message.Body)) {
        case "Half life":
            await processMessageEventArgs.AbandonMessageAsync(message);
            await Error.WriteLineAsync("Abandon half life message");
            break;

        case "Delivery Count":
            await Error.WriteLineAsync("Throwing delivery count message");
            throw new InvalidOperationException();

        case "Poor Soul":
            await Error.WriteLineAsync("Dead letter poor soul message");
            await processMessageEventArgs.DeadLetterMessageAsync(message,
                new Dictionary<string, object>
                {
                    {"Reason", "Because we can!"},
                    {"When", DateTimeOffset.UtcNow}
                });
            break;
    }
};
```

```
c:\p\AzureServiceBus.DeepDive\Deadlettering
> dotnet run
Sent half life message
Sent delivery count message
Sent poor soul message
Throwing delivery count message
Dead letter poor soul message
```

Service Bus Namespace

- sb://nservicebus-tests.servicebus.windows.net/
 - Queues
 - Topics
 - Event Hubs
 - Notification Hub
 - Relays

queue (0, 3, 0)

- Set Status
- Delete Queue
- Refresh Queue F5
- Rename Queue
- Export Queue
- Copy Queue Url
- Copy Deadletter Queue Url
- Test Queue In SDI Mode
- Test Queue In MDI Mode
- Send Messages
- Create Queue Listener
- Receive Messages
- Receive Deadletter Queue Messages
- Receive Transfer Deadletter Queue Messages
- Purge Messages
- Purge Deadletter Queue Messages

View Queue: queue

Description Authorization Rules Messages

Message List

MessageId	Seq	Size	Label	EnqueuedTimeUtc	ExpiresAtUtc

Message System Properties

Message Text

1

Message Custom Properties

Name	Value

Purge Purge DLQ Messages Deadletter Transf DLQ Refresh Status Delete Update

Log

<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.
<22:42:19> 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.
<22:44:42> The queue queue has been successfully retrieved.
<22:44:50> [0] messages peeked from the queue [queue].
<23:11:32> The queue queue has been successfully retrieved.
<23:12:18> The queue queue has been successfully retrieved.
<23:12:20> The queue queue has been successfully retrieved.

Service Bus Explorer

File Edit Actions View Help

Microsoft Azure

Service Bus Namespace

- sb://nservicebus-tests.servicebus.windows.net/
 - Queues
 - queue (0, 3, 0)
 - Topics
 - Event Hubs
 - Notification Hubs
 - Relays

View Queue: queue

Description Authorization Rules Messages Deadletter

Message List

MessageId	Seq	Size	Label	EnqueuedTimeUtc	ExpiresAtUtc
e1139faea7445249e520e...	1	227		14.06.2019 21:12	14.06.2019 21:12
dfc3c370cba04145b4d89e...	2	270		14.06.2019 21:12	31.12.9999 23:59
cc10d022cceb4944b359e1...	3	207		14.06.2019 21:12	31.12.9999 23:59

Message System Properties

Message Text

Delivery Count

Message Custom Properties

Name	Value
DeadLetterRea...	MaxDeliveryCountExceeded
DeadLetterError...	Message could not be consumed after 1 delivery atte...

Purge Purge DLQ Messages Deadletter Transf DLQ Refresh Status Delete Update

Log

<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.
<22:42:19> The queue queue has been successfully retrieved.
<22:44:42> The queue queue has been successfully retrieved.
<22:44:50> [0] messages peeked from the queue [queue].
<23:11:32> The queue queue has been successfully retrieved.
<23:12:18> The queue queue has been successfully retrieved.
<23:12:20> The queue queue has been successfully retrieved.
<23:12:26> [3] messages peeked from the deadletter queue of the queue [queue].

Service Bus Explorer

File Edit Actions View Help Microsoft Azure

Service Bus Namespace

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

- Queues
 - queue (0, 3, 0)
- Topics
- Event Hubs
- Notification Hubs
- Relays

View Queue: queue

Description Authorization Rules Messages Deadletter

Message List

MessageId	Seq	Size	Label	EnqueuedTimeUtc	ExpiresAtUtc
e11399aea7445249e520e...	1	227		14.06.2019 21:12	14.06.2019 21:12
dfc3c370cba04145b4d89e...	2	270		14.06.2019 21:12	31.12.9999 23:59
cc10d022cceb4944b359e1...	3	207		14.06.2019 21:12	31.12.9999 23:59

Message System Properties

Message Text

1 Delivery Count

Message Custom Properties

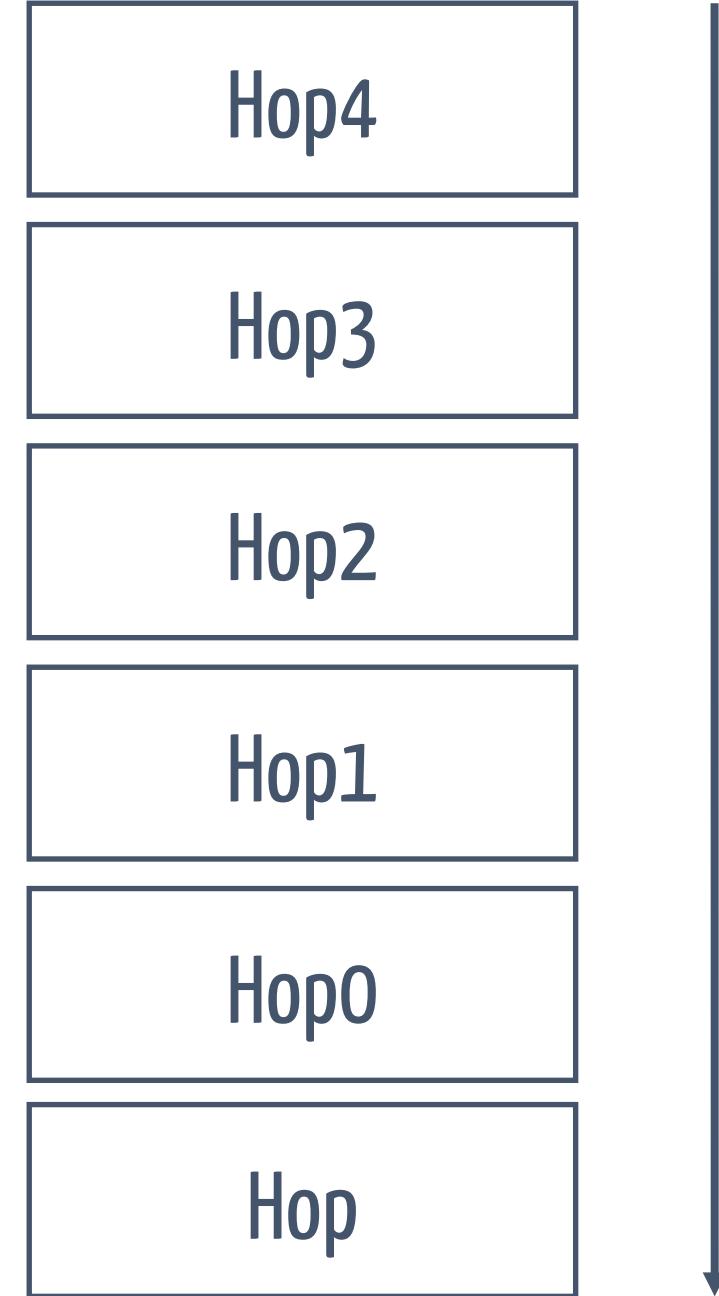
Name	Value
DeadLetterRe...	MaxDeliveryCountExceeded
DeadLetterError...	Message could not be consumed after 1 delivery atte...

Purge Purge DLQ Messages Deadletter Transf DLQ Refresh Status Delete Update

Log

```
<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.  
<22:42:19> The queue queue has been successfully retrieved.  
<22:44:42> The queue queue has been successfully retrieved.  
<22:44:50> [0] messages peeked from the queue [queue].  
<23:11:32> The queue queue has been successfully retrieved.  
<23:12:18> The queue queue has been successfully retrieved.  
<23:12:20> The queue queue has been successfully retrieved.  
<23:12:26> [3] messages peeked from the deadletter queue of the queue [queue].
```

Forwarding



ForwardTo



```
var description = new CreateQueueOptions("Hop");
await client.CreateQueueAsync(description);

description = new CreateQueueOptions("Hop0");
await client.CreateQueueAsync(description);

description = new CreateQueueOptions("Hop1")
{
    ForwardTo = "Hop0"
};
await client.CreateQueueAsync(description);

description = new CreateQueueOptions("Hop2")
{
    ForwardTo = "Hop1"
};
await client.CreateQueueAsync(description);

description = new CreateQueueOptions("Hop3")
{
    ForwardTo = "Hop2"
};
await client.CreateQueueAsync(description);

description = new CreateQueueOptions("Hop4")
{
    ForwardTo = "Hop3"
};
await client.CreateQueueAsync(description);
```

```
await using var serviceBusClient =
    new ServiceBusClient(connectionString);
var sender = serviceBusClient.CreateSender("Hop4");

var message = new ServiceBusMessage("Weeeeeeehhh!");
await sender.SendMessageAsync(message);
```

```
c:\p\AzureServiceBus.DeepDive\Forwarding
> dotnet run
Sent message
Got 'Weeeeeeehhh!' on hop 'Hop0'
Setup forwarding from Hop0 to Hop
```

Service Bus Namespace

sb://nservicebustests.servicebus.windows.net/
Queues
↳ hop (0, 0, 0)
↳ hop0 (0, 0, 0)
↳ hop1 (0, 0, 0)
↳ hop2 (0, 0, 0)
↳ hop3 (0, 0, 0)
↳ hop4 (0, 0, 0)
Topics
Event Hubs
Notification Hubs
Relays

Entity**Log**

```
<23:33:35> The queue hop has been successfully retrieved.  
<23:33:35> The queue hop0 has been successfully retrieved.  
<23:33:35> The queue hop1 has been successfully retrieved.  
<23:33:35> The queue hop2 has been successfully retrieved.  
<23:33:35> The queue hop3 has been successfully retrieved.  
<23:33:35> The queue hop4 has been successfully retrieved.
```

Service Bus Namespace

sb://nservicebustests.servicebus.windows.net/
Queues
hop (0, 0, 0)
hop0 (0, 0, 0)
hop1 (0, 0, 0)
hop2 (0, 0, 0)
hop3 (0, 0, 0)
hop4 (0, 0, 0)
Topics
Event Hubs
Notification Hubs
Relays

View Queue: hop0**Description** | Authorization Rules |**Path**

Relative URI:

hop0

Auto Delete On Idle

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

Duplicate Detection History Time Window

Days:	Hours:	Minutes:	Seconds:	Millisecs:
0	0	10	0	0

Default Message Time To Live

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

Queue Properties

Max Queue Size In GB:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------

User Description

hop

Forward To

hop

...

Forward Dead Lettered Messages To

hop

...

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

Purge **Purge DLQ** **Messages** **Deadletter** **Transf DLQ** **Refresh** **Status** **Delete****Log**

<23:33:35> The queue hop has been successfully retrieved.
<23:33:35> The queue hop0 has been successfully retrieved.
<23:33:35> The queue hop1 has been successfully retrieved.
<23:33:35> The queue hop2 has been successfully retrieved.
<23:33:35> The queue hop3 has been successfully retrieved.
<23:33:35> The queue hop4 has been successfully retrieved.

```
c:\p\AzureServiceBus.DeepDive\Forwarding
> dotnet run
Sent message
Got 'Weeeeeeehh!' on hop 'Hop0'
Setup forwarding from Hop0 to Hop
```

Service Bus Namespace

sb://hservicebustests.servicebus.windows.net/

- Queues
 - hop (0, 0, 0)
 - hop0 (0, 1, 0) (Selected)
 - hop1 (0, 0, 0)
 - hop2 (0, 0, 0)
 - hop3 (0, 0, 0)
 - hop4 (0, 0, 0)
- Topics
- Event Hubs
- Notification Hubs
- Relays

View Queue: hop0

Description | Authorization Rules |

Path

Relative URI: hop0

Auto Delete On Idle

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

Duplicate Detection History Time Window

Days:	Hours:	Minutes:	Seconds:	Millisecs:
0	0	10	0	0

Default Message Time To Live

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

Queue Properties

Max Queue Size In GB: 1 GB

Max Delivery Count: 10

User Description:

Forward To: hop

Forward Dead Lettered Messages To:

Lock Duration

Days:	Hours:	Minutes:	Seconds:	Millisecs:
0	0	1	0	0

Queue Settings

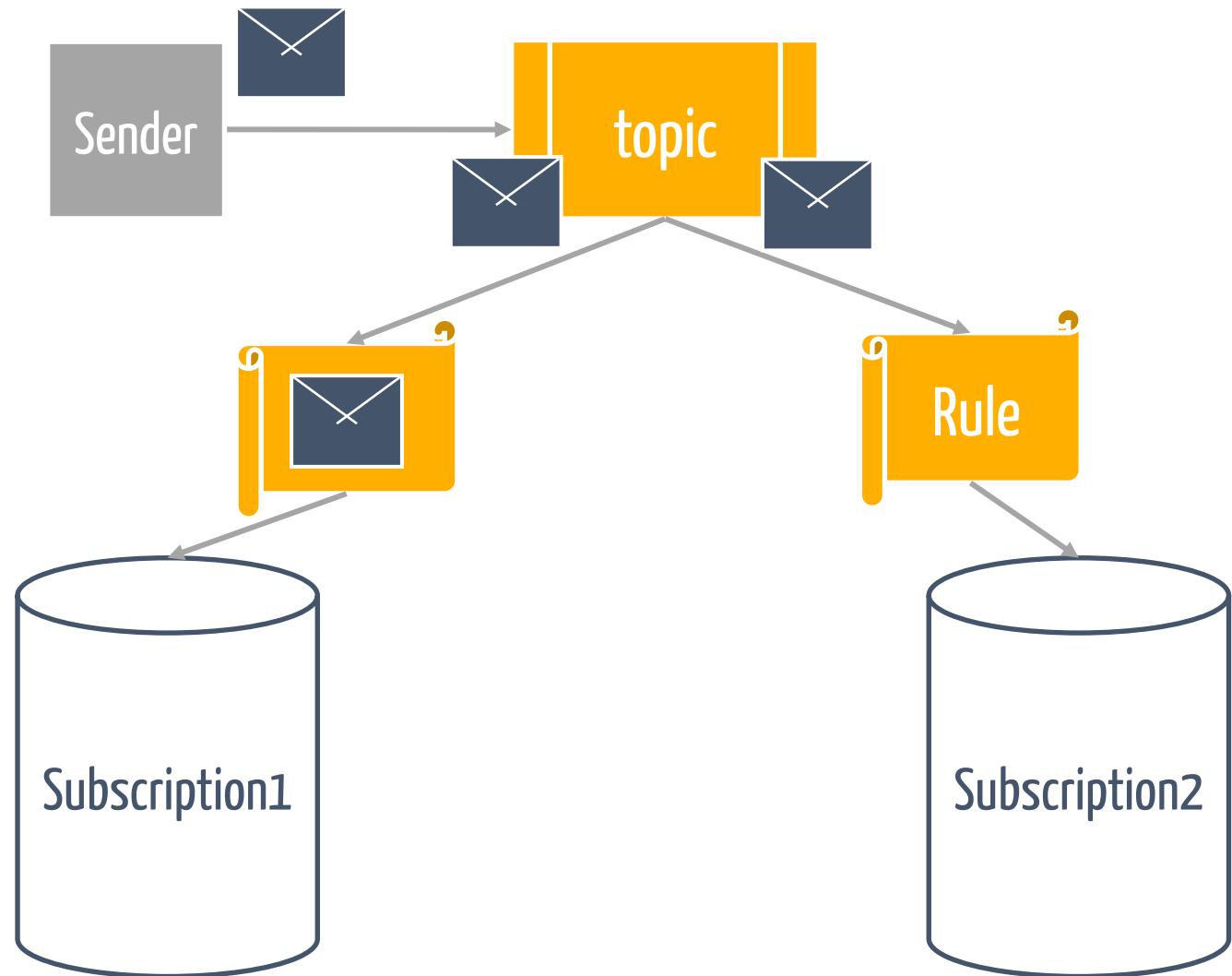
Enable Batched Operations
 Enable Dead Lettering On Message Expiration
 Enable Partitioning
 Enable Express
 Requires Duplicate Detection
 Requires Session
 Enforce Message Ordering

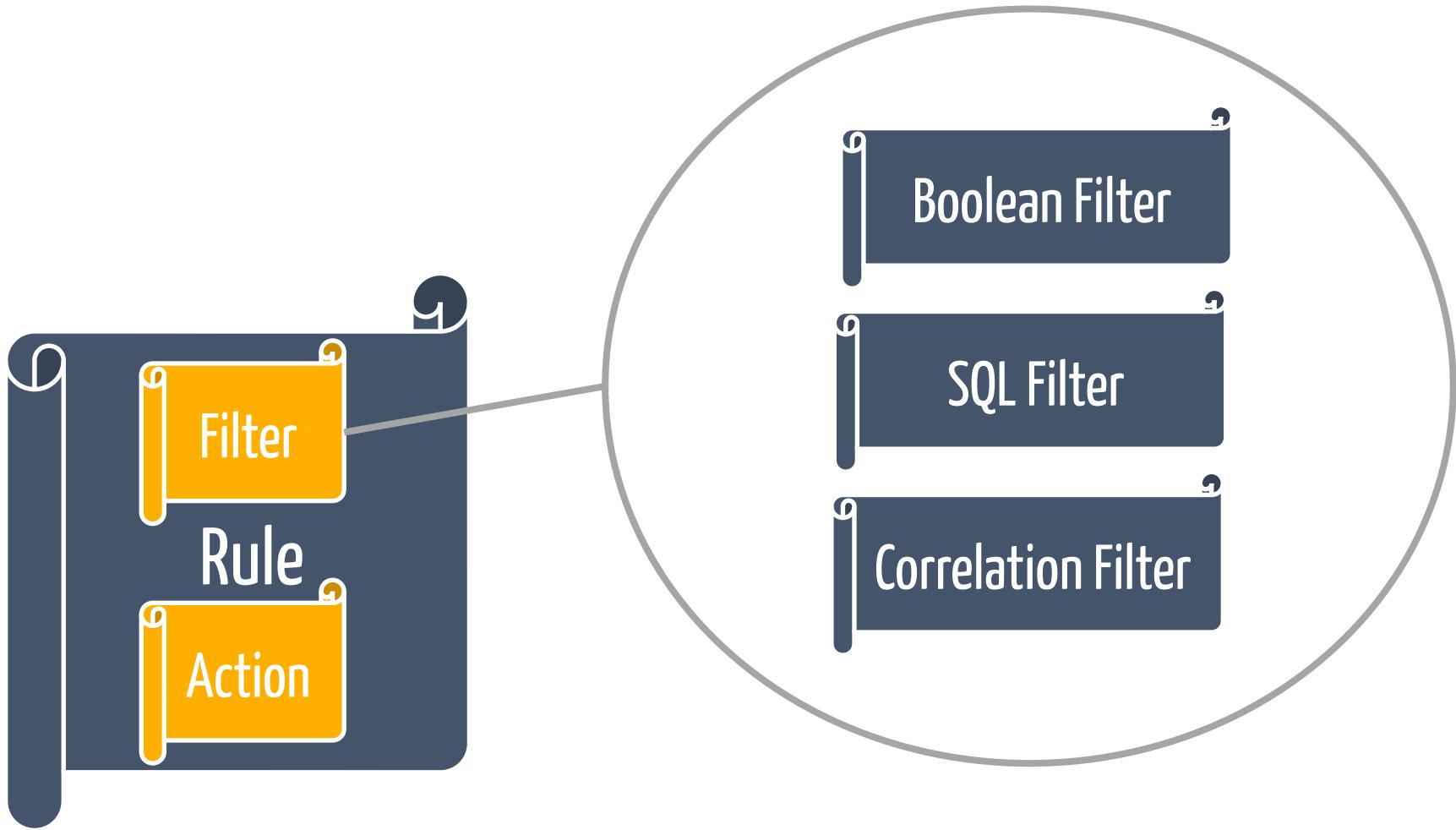
Purge Purge DLQ Messages Deadletter Transfer DLQ Refresh Status Delete Up

Log

```
<23:33:35> The queue hop has been successfully retrieved.  
<23:33:35> The queue hop0 has been successfully retrieved.  
<23:33:35> The queue hop1 has been successfully retrieved.  
<23:33:35> The queue hop2 has been successfully retrieved.  
<23:33:35> The queue hop3 has been successfully retrieved.  
<23:33:35> The queue hop4 has been successfully retrieved.  
<23:33:45> The queue hop0 has been successfully updated.  
<23:34:10> The queue hop has been successfully retrieved.  
<23:34:10> The queue hop0 has been successfully retrieved.  
<23:34:10> The queue hop1 has been successfully retrieved.  
<23:34:10> The queue hop2 has been successfully retrieved.  
<23:34:10> The queue hop3 has been successfully retrieved.  
<23:34:10> The queue hop4 has been successfully retrieved.
```

Pub/Sub





```
var topicDescription = new CreateTopicOptions(topicName);
await client.CreateTopicAsync(topicDescription);

var subscriptionDescription =
    new CreateSubscriptionOptions(topicName, rushSubscription);
await client.CreateSubscriptionAsync(subscriptionDescription);

subscriptionDescription =
new CreateSubscriptionOptions(topicName, currencySubscription);
await client.CreateSubscriptionAsync(subscriptionDescription);
```

```
var ruleDescription = new CreateRuleOptions
{
    Name = "MessagesWithRushlabel",
    Filter = new CorrelationRuleFilter
    {
        Subject = "rush"
    },
    Action = null
};
await client.
    CreateRuleAsync(topicName, rushSubscription, ruleDescription);
```

```
ruleDescription = new CreateRuleOptions
{
    Name = "MessagesWithCurrencyCHF",
    Filter = new SqlRuleFilter("currency = 'CHF'"),
    Action = new SqlRuleAction("SET currency = 'Złoty'")
};
await client
    .CreateRuleAsync(topicName, currencySubscription, ruleDescription);
```

```
var message = new ServiceBusMessage("Damn I have no time!")
{
    Subject = "rush"
};
await sender.SendMessageAsync(message);

message = new ServiceBusMessage("I'm rich! I have 1000");
message.ApplicationProperties.Add("currency", "CHF");
await sender.SendMessageAsync(message);
```

View Subscription: alwaysInRush

Description

Name
Subscription Name: alwaysInRush

Auto Delete On Idle
Days: 106751 Hours: 2 Minutes: 48 Seconds: 5 Millisecs: 477

Lock Duration
Days: 0 Hours: 0 Minutes: 1 Seconds: 0 Millisecs: 0

Default Message Time To Live
Days: 106751 Hours: 2 Minutes: 48 Seconds: 5 Millisecs: 477

Subscription Properties

Max Delivery Count: 10
User Description:

Subscription Settings

Enable Batched Operations
 Enable Dead Lettering On Filter Evaluation Error
 Enable Dead Lettering On Message Expiration
 Requires Session

Subscription Information

Name	Value
Status	Active
Is ReadOnly	False
Created At	17.06.2019 10:35:53
Accessed At	17.06.2019 10:35:53
Updated At	17.06.2019 10:35:53
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 Refresh Disable Delete Update

Log

```
<12:36:41> The file C:\p\AzureServiceBus.Deep Dive\explorer\ServiceBusExplorer.exe.Config is used for the configuration settings.  
<12:36:45> The application is now connected to the sb://nsericebustests.servicebus.windows.net/ service bus namespace.  
<12:36:45> MessagingFactory successfully created  
<12:36:48> The topic topic has been successfully retrieved.  
<12:36:48> The subscription alwaysInRush for the topic topic has been successfully retrieved.  
<12:36:48> The subscription maybeRich for the topic topic has been successfully retrieved.  
<12:37:46> The rule MessagesWithRushlabel for the alwaysInRush subscription of the topic topic has been successfully retrieved.
```

Service Bus Explorer

File Edit Actions View Help Microsoft Azure

Service Bus Namespace

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

- Queues
- Topics
 - topic
 - Subscriptions
 - alwaysInRush (1, 0, 0)
 - maybeRich (1, 0, 0)
- Event Hubs
- Notification Hubs
- Relays

View Subscription: maybeRich

Description

Name

Subscription Name: maybeRich

Auto Delete On Idle

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

Lock Duration

Days: 0 Hours: 0 Minutes: 1 Seconds: 0 Millisecs: 0

Default Message Time To Live

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

Subscription Properties

Max Delivery Count: 10

User Description:

Forward To:

Forward Dead Lettered Messages To:

Subscription Settings

Enable Batched Operations
 Enable Dead Lettering On Filter Evaluation Error
 Enable Dead Lettering On Message Expiration
 Requires Session

Subscription Information

Name	Value
Status	Active
Is ReadOnly	False
Created At	17.06.2019 10:35:53
Accessed At	17.06.2019 10:35:53
Updated At	17.06.2019 10:35:53
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 Refresh Disable Delete Update

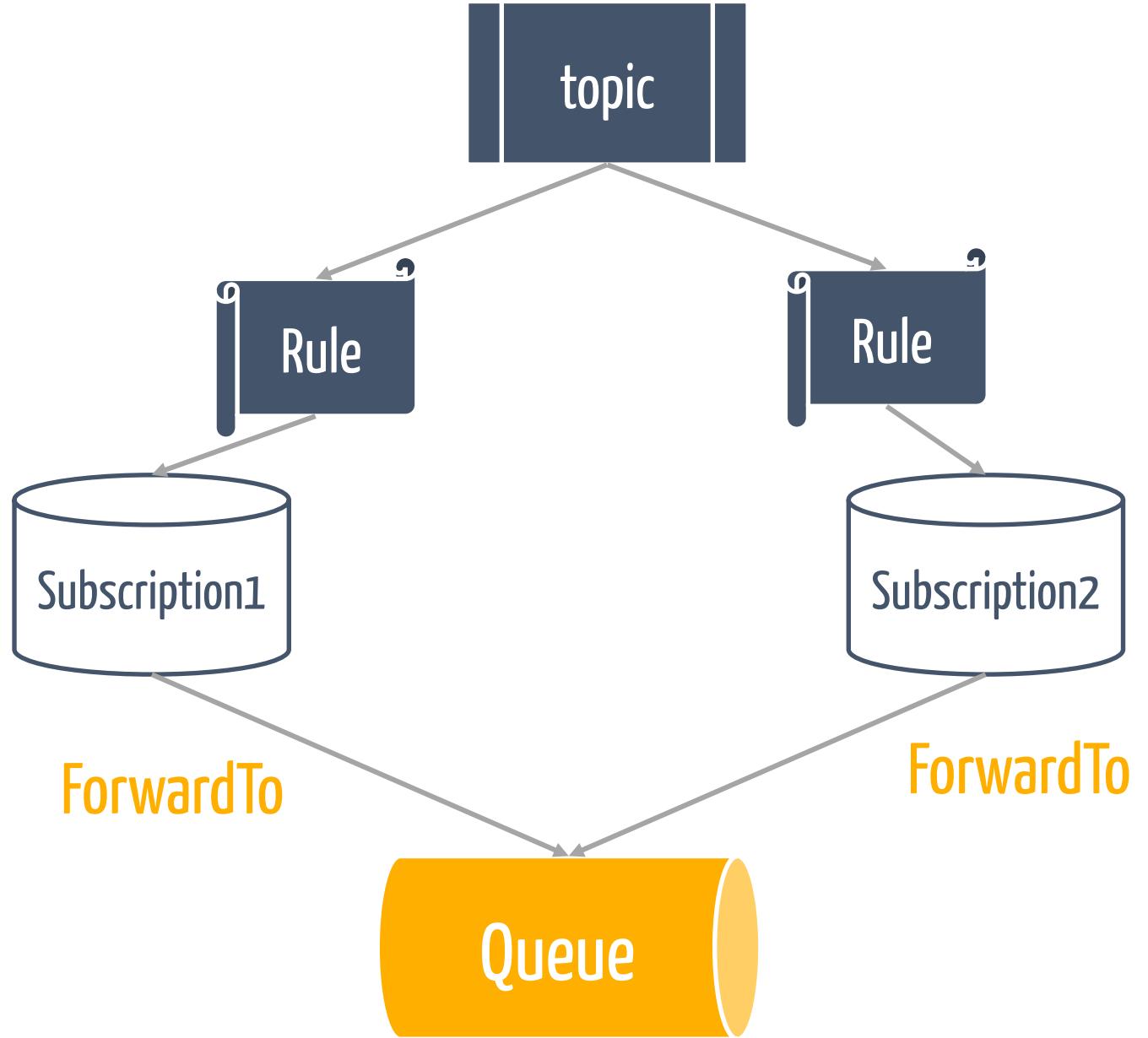
Log

```
<12:36:41> The file C:\p\AzureServiceBus.DeepDive\explorer\ServiceBusExplorer.exe.Config is used for the configuration settings.
<12:36:45> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.
<12:36:45> MessagingFactory successfully created
<12:36:48> The topic topic has been successfully retrieved.
<12:36:48> The subscription alwaysInRush for the topic topic has been successfully retrieved.
<12:36:48> The subscription maybeRich for the topic topic has been successfully retrieved.
<12:37:46> The rule MessagesWithRushLabel for the alwaysInRush subscription of the topic topic has been successfully retrieved.
<12:37:54> [1] messages peeked from the subscription [alwaysInRush].
<12:38:01> The rule MessagesWithCurrencyCHF for the maybeRich subscription of the topic topic has been successfully retrieved.
```

Favor Correlation filter over SQL filter

Subscriptions are virtual queues and
subscribers need to receive from them

Topologies



```
var subscriptionDescription =
    new CreateSubscriptionOptions(topicName, rushSubscription)
{
    ForwardTo = inputQueue
};
await client.CreateSubscriptionAsync(subscriptionDescription);

subscriptionDescription =
new CreateSubscriptionOptions(topicName, currencySubscription)
{
    ForwardTo = inputQueue
};
await client.CreateSubscriptionAsync(subscriptionDescription);
```

cmd - dotnet r...

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

Service Bus Explorer

File Edit Actions View Help

Microsoft Azure

Service Bus Namespace

- sb://nservicebustests.servicebus.windows.net/
 - Queues
 - queue (2, 0, 0)
 - Topics
 - topic
 - Subscriptions
 - alwaysInRush (0, 0, 0)
 - maybeRich (0, 0, 0)
 - Event Hubs
 - Notification Hubs
 - Relays

View Topic: topic

Description | Authorization Rules

Path
Relative URI: topic

Auto Delete On Idle
Days: 106751 Hours: 2 Minutes: 48 Seconds: 5 Millisecs: 477

Default Message Time To Live
Days: 106751 Hours: 2 Minutes: 48 Seconds: 5 Millisecs: 477

Topic Properties
Max Queue Size In GB: 1 GB

Duplicate Detection History Time Window
Days: 0 Hours: 0 Minutes: 10 Seconds: 0 Millisecs: 0

Topic Settings
 Enable Batched Operations
 Enable Filtering Messages Before Publishing
 Enable Partitioning
 Enable Express
 Requires Duplicate Detection
 Enforce Message Ordering

Topic Information

Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	0
Created At	17.06.2019 10:52:29
Accessed At	17.06.2019 10:52:33
Updated At	17.06.2019 10:52:29
Active Message Count	0
DeadLetter Message Count	0
Scheduled Message Count	0
Transfer Message Count	0
Transfer DL Message Count	0

Refresh Disable Delete Update

Log

```
<12:36:41> The file C:\p\AzureServiceBus.DeepDive\explorer\ServiceBusExplorer.exe.Config is used for the configuration settings.
<12:36:45> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.
<12:36:45> MessagingFactory successfully created
<12:36:48> The topic topic has been successfully retrieved.
<12:36:48> The subscription alwaysInRush for the topic topic has been successfully retrieved.
<12:36:48> The subscription maybeRich for the topic topic has been successfully retrieved.
<12:37:46> The rule MessagesWithRushLabel for the alwaysInRush subscription of the topic topic has been successfully retrieved.
<12:37:54> [1] messages peeked from the subscription [alwaysInRush].
<12:38:01> The rule MessagesWithCurrencyCHF for the maybeRich subscription of the topic topic has been successfully retrieved.
<12:38:04> [1] messages peeked from the subscription [maybeRich].
<12:53:06> The queue queue has been successfully retrieved.
<12:53:06> The topic topic has been successfully retrieved.
<12:53:07> The subscription alwaysInRush for the topic topic has been successfully retrieved.
<12:53:07> The subscription maybeRich for the topic topic has been successfully retrieved.
```

12:56
17.06.2019

Service Bus Explorer

File Edit Actions View Help

Microsoft Azure

Service Bus Namespace

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

- Queues
 - queue (2, 0, 0)
- Topics
 - topic
 - Subscriptions
 - alwaysInRush (0, 0, 0)
 - maybeRich (0, 0, 0)
- Event Hubs
- Notification Hubs
- Relays

View Queue: queue

Description | Authorization Rules | Messages |

Message List

MessageId	Seq	Size	Label	EnqueuedTimeUtc	ExpiresAtUtc
6b1fe11b26be418ab3b349f...	1	228	rush	17.06.2019 10:52	31.12.9999 23:59
ccbbea7401af434890a2f38...	2	273		17.06.2019 10:52	31.12.9999 23:59

Message System Properties

Misc	
ContentType	
CorrelationId	
DeadLetterSource	
DeliveryCount	1
EnqueuedSequenceNumber	2
EnqueuedTimeUtc	17.06.2019 10:52
ExpiresAtUtc	31.12.9999 23:59
ForcePersistence	False
IsBodyConsumed	False
Label	
LockedUntilUtc	
LockToken	
MessageId	ccbbea7401af434890a2f381007

Message Text

```
I'm rich! I have 1000
```

Message Custom Properties

Name	Value
currency	Zloty
RuleName	MessagesWithCurrencyCHF

Purge | Purge DLQ | Messages | Deadletter | Transf DLQ | Refresh | Status | Delete | Update

Log

```
<12:36:45> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.  
<12:36:45> MessagingFactory successfully created  
<12:36:48> The topic topic has been successfully retrieved.  
<12:36:48> The subscription alwaysInRush for the topic topic has been successfully retrieved.  
<12:36:48> The subscription maybeRich for the topic topic has been successfully retrieved.  
<12:37:46> The rule MessagesWithRushlabel for the alwaysInRush subscription of the topic topic has been successfully retrieved.  
<12:37:54> [1] messages peeked from the subscription [alwaysInRush].  
<12:38:01> The rule MessagesWithCurrencyCHF for the maybeRich subscription of the topic topic has been successfully retrieved.  
<12:38:04> [1] messages peeked from the subscription [maybeRich].  
<12:53:06> The queue queue has been successfully retrieved.  
<12:53:06> The topic topic has been successfully retrieved.  
<12:53:07> The subscription alwaysInRush for the topic topic has been successfully retrieved.  
<12:53:07> The subscription maybeRich for the topic topic has been successfully retrieved.  
<12:53:13> The rule MessagesWithRushlabel for the alwaysInRush subscription of the topic topic has been successfully retrieved.  
<12:53:15> The rule MessagesWithCurrencyCHF for the maybeRich subscription of the topic topic has been successfully retrieved.  
<12:53:24> [2] messages peeked from the queue [queue].
```



Atomic Sends

```
using (var scope =
    new TransactionScope(TransactionScopeAsyncFlowOption.Enabled))
{
    var message = new ServiceBusMessage("Deep Dive 1");
    await sender.SendMessageAsync(message);

    message = new ServiceBusMessage("Deep Dive 2");
    await sender.SendMessageAsync(message);

    scope.Complete();
}
```

```
c:\p\AzureServiceBus.DeepDive\AtomicSend
> dotnet run
Sent message 1 in transaction '12c87d8b-87ad-4816-8218-7225fe9f850e:1'
#'0' messages in 'queue'
Sent message 2 in transaction '12c87d8b-87ad-4816-8218-7225fe9f850e:1'
About to complete transaction scope.
```

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

```
> dotnet run
```

```
Sent message 1 in transaction '12c87d8b-87ad-4816-8218-7225fe9f850e:1'
```

```
'0' messages in 'queue'
```

```
Sent message 2 in transaction '12c87d8b-87ad-4816-8218-7225fe9f850e:1'
```

```
About to complete transaction scope.
```

```
'0' messages in 'queue'
```

```
Completed transaction scope.
```

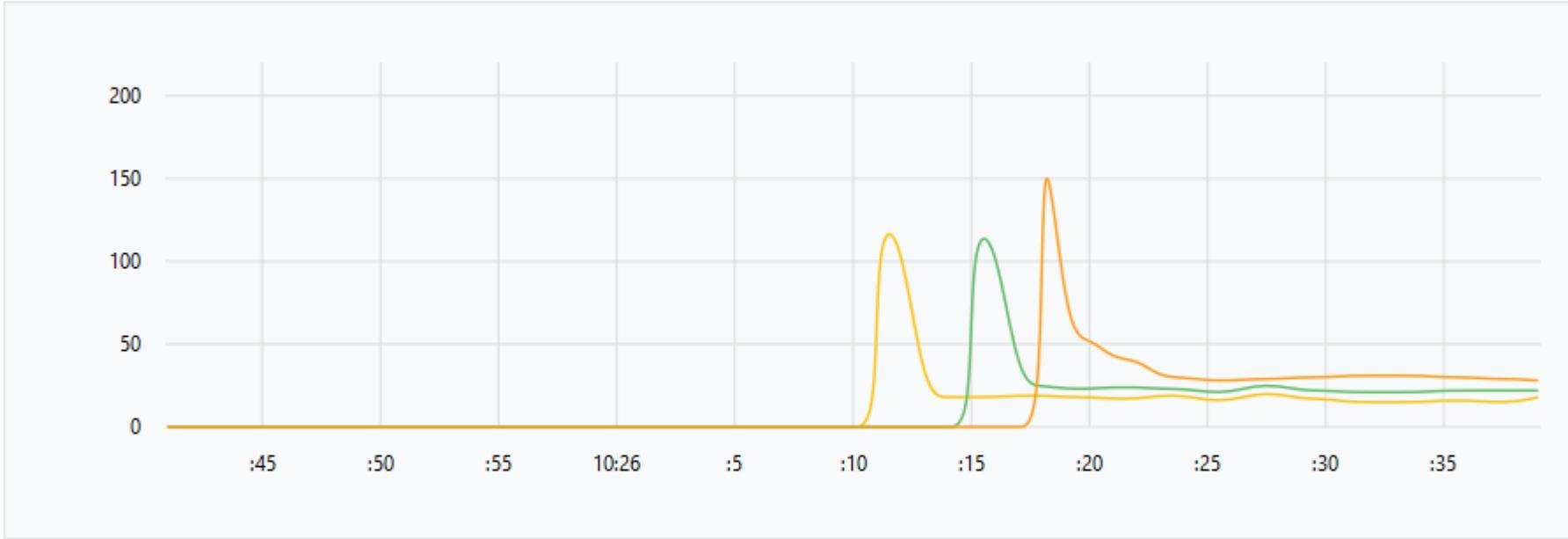
```
'2' messages in 'queue'
```

Transaction not completed

Transaction completed

Batching

Region	Average Latency (ms)
Switzerland North (Zurich)	18 ms
Germany West Central (Frankfurt)	22 ms
France Central (Paris)	28 ms



Latency Test

Geography	Region	Physical Location	Average Latency (ms)
Europe	France Central	Paris	28 ms
Europe	Germany West Central	Frankfurt	22 ms
Europe	Switzerland North	Zurich	18 ms

```
var messages = new List<ServiceBusMessage>();
for (var i = 0; i < 10; i++)
{
    var message = new ServiceBusMessage("Deep Dive{i}");
    messages.Add(message);
}

await sender.SendMessagesAsync(messages);
```

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

```
> dotnet run
```

```
Sending 10 messages in a batch.
```

```
Sending 6500 messages in a batch.
```

```
The received message (delivery-id:0, size:271890 bytes) exceeds the limit (262144 bytes) currently allowed on the link.
```

```
Sending 101 messages in a batch with in transaction 'e943ed70-447a-421c-a777-876b2249bf9a:1'.
```

```
var messagesToSend = new Queue<ServiceBusMessage>();
for (var i = 0; i < 4500; i++)
{
    var message = new ServiceBusMessage($"Deep Dive{i}. Deep Dive{i}. Deep Dive{i}.");
    messagesToSend.Enqueue(message);
}
```

```
var messageCount = messagesToSend.Count;
int batchCount = 1;
while (messagesToSend.Count > 0)
{
    using ServiceBusMessageBatch messageBatch = await sender.CreateMessageBatchAsync();

    if (messageBatch.TryAddMessage(messagesToSend.Peek()))
    {
        messagesToSend.Dequeue();
    }
    else
    {
        throw new Exception($"Message {messageCount - messagesToSend.Count} is too large a
nd cannot be sent.");
    }

    while (messagesToSend.Count > 0 && messageBatch.TryAddMessage(messagesToSend.Peek()))
    {
        messagesToSend.Dequeue();
    }

    await sender.SendMessagesAsync(messageBatch);
}
```

Upgrade to premium tier

Use Claim Check Pattern

Send Via

Incoming



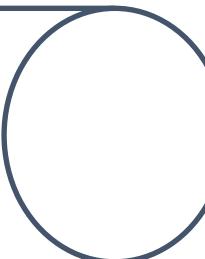
Transfer



Outgoing1



Outgoing2



```
await using var transactionalClient =
    new ServiceBusClient(connectionString,
    new ServiceBusClientOptions
{
    EnableCrossEntityTransactions = true,
});

receiver = transactionalClient.CreateProcessor(inputQueue);
sender = transactionalClient.CreateSender(destinationQueue);
```

```
receiver.ProcessMessageAsync += async processMessageEventArgs =>
{
    var message = processMessageEventArgs.Message;
    using (var scope = new TransactionScope(TransactionScopeAsyncFlowOption.Enabled))
    {
        await sender.SendMessageAsync(new ServiceBusMessage("Will not leak"));

        if (!message.ApplicationProperties.ContainsKey("Win"))
            throw new InvalidOperationException();

        await sender.SendMessageAsync(new ServiceBusMessage("Will not leak"));

        scope.Complete();
    }
};
```

```
c:\p\AzureServiceBus.DeepDive\SendVia
> dotnet run
#'0' messages in 'destination'
Received message with 'bdc126e8424549acb9d1c23a5af31799' and content 'Kick off'
```

<1> cmd - dotnet r...

```
c:\p\AzureServiceBus.DeepDive\SendVia
> dotnet run
#'0' messages in 'destination'
Received message with 'bdc126e8424549acb9d1c23a5af31799' and content 'Kick off'
#'1' messages in 'destination'
Received message with 'bdc126e8424549acb9d1c23a5af31799' and content 'Kick off'
#'2' messages in 'destination'

Received message with '3ef5f1087b43424ba0fcf1987df53ee5' and content 'Fail'
#'0' messages in 'destination'
Received message with '3ef5f1087b43424ba0fcf1987df53ee5' and content 'Fail'
#'0' messages in 'destination'
Received message with 'b2f109763d3442788e6cf41f67fce078' and content 'Win'
#'2' messages in 'destination'
```

TransferDLQ

```
receiver.ProcessMessageAsync += async processMessageEventArgs =>
{
    var message = processMessageEventArgs.Message;
    using (var scope = new TransactionScope(TransactionScopeAsyncFlowOption.Enabled))
    {
        await sender.SendMessageAsync(new ServiceBusMessage("Will not leak"));

        var client = new ServiceBusAdministrationClient(connectionString);

        QueueProperties queueProperties = await client.GetQueueAsync(destinationQueue);
        queueProperties.Status = EntityStatus.SendDisabled;

        await client.UpdateQueueAsync(queueProperties);

        scope.Complete();
    }
};
```

```
var client = new ServiceBusAdministrationClient(connectionString);

QueueRuntimeProperties info = await client
    .GetQueueRuntimeInfoAsync(destination);

long activeMessageCount = info.ActiveMessageCount;
long deadLetterMessageCount = info.DeadLetterMessageCount;
long transferDeadLetterMessageCount = info.TransferDeadLetterMessageC
ount;

string destinationDeadLetterPath = EntityNameHelper
    .FormatDeadLetterPath(destination);
string destinationTransferDeadLetterPath = EntityNameHelper
    .FormatTransferDeadLetterPath(destination);
```

<1> cmd - dotnet r...

```
c:\p\AzureServiceBus.DeepDive\TransferDLQ
> dotnet run
Received message with '4509ba5ef4f94bcdbfa1972a56eed23d' and content 'Kick off'
#'1' messages in 'queue'
#'0' messages in 'queue/$DeadLetterQueue'
#'1' messages in 'queue/$Transfer/$DeadLetterQueue'
```

Dedup

```
var queueDescription = new CreateQueueOptions(destination)
{
    RequiresDuplicateDetection = true,
    DuplicateDetectionHistoryTimeWindow = TimeSpan.FromSeconds(20)
};
await client.CreateQueueAsync(queueDescription);
```

```
var content = Encoding.UTF8.GetBytes("Message1Message1");
var messageId = new Guid(content).ToString();

var messages = new List<ServiceBusMessage>
{
    new ServiceBusMessage(content) { MessageId = messageId },
    new ServiceBusMessage(content) { MessageId = messageId },
    new ServiceBusMessage(content) { MessageId = messageId }
};

await sender.SendMessagesAsync(messages);
```

cmd - dotnet run

<1> cmd - dotnet r...

Search

```
c:\p\AzureServiceBus.DeepDive\Dup  
> dotnet run  
Messages sent  
Received message with '7373654d-6761-3165-4d65-737361676531' and content 'Message1Message1'
```

SUMMARY

ENTERPRISE MESSAGING FEATURES

MESSAGE TRANSACTIONAL PROCESSING

RELIABILITY

GUARANTEED THROUGHPUT AND LATENCY

TOTAL: PRICELESS



Azure Service Bus Transport

Tags ▾

Source | NServiceBus.Transport.AzureServiceBus (1.x)

Target NServiceBus Version: 7.x

The Azure Service Bus transport leverages the .NET Standard Microsoft.Azure.ServiceBus[®] client SDK.

Azure Service Bus[®] is a messaging service hosted on the Azure platform that allows for exchanging messages between various applications in a loosely coupled fashion. The service offers guaranteed message delivery and supports a range of standard protocols (e.g. REST, AMQP, WS*) and APIs such as native pub/sub, delayed delivery, and more.

Configuring

go.particular.net/Sample

To use Azure Service Bus as the underlying transport:

```
var transport = endpointConfiguration.UseTransport<AzureServiceBusTransport>();  
transport.ConnectionString("Endpoint=sb://[NAMESPACE].servicebus.windows.net/;SharedAccessKeyName=[KEYNAME];Shared  
AccessKey=[KEY]");
```

Copy/Edit ▾

The Azure Service Bus transport requires a connection string to connect to a namespace.

Samples

- Azure Service Bus Native Integration
Consuming messages published by non-NServiceBus endpoints.
- Azure Service Bus Send/Reply Sample
Demonstrates the send/reply pattern with Azure Service Bus.

Getting Started

NServiceBus

Transports

General

Upgrade Guides

Azure Service Bus

Configuration

Backwards Compatibility

Transaction Support

Operational Scripting

Azure Service Bus (Legacy)

Azure Storage Queues

SQL Server

MSMQ

RabbitMQ

Amazon SQS



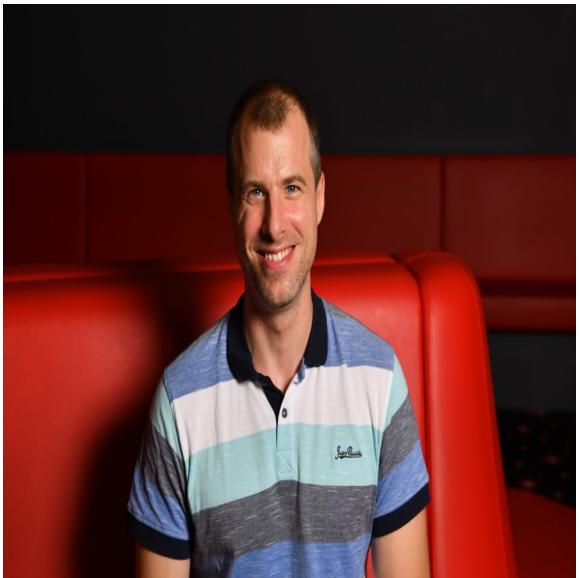
Thanks

Slides, Links...

github.com/danielmarbach/AzureServiceBus.DeepDive



Q & A



Software Engineer
Microsoft MVP

@danielmarbach
particular.net/blog
planetgeek.ch