Particular
Software

Deep Dive with Azure Service Bus

@danielmarbach

Events

Event Grid

Cross cloud reactive eventing

Event Hubs

Big data streaming

Messages

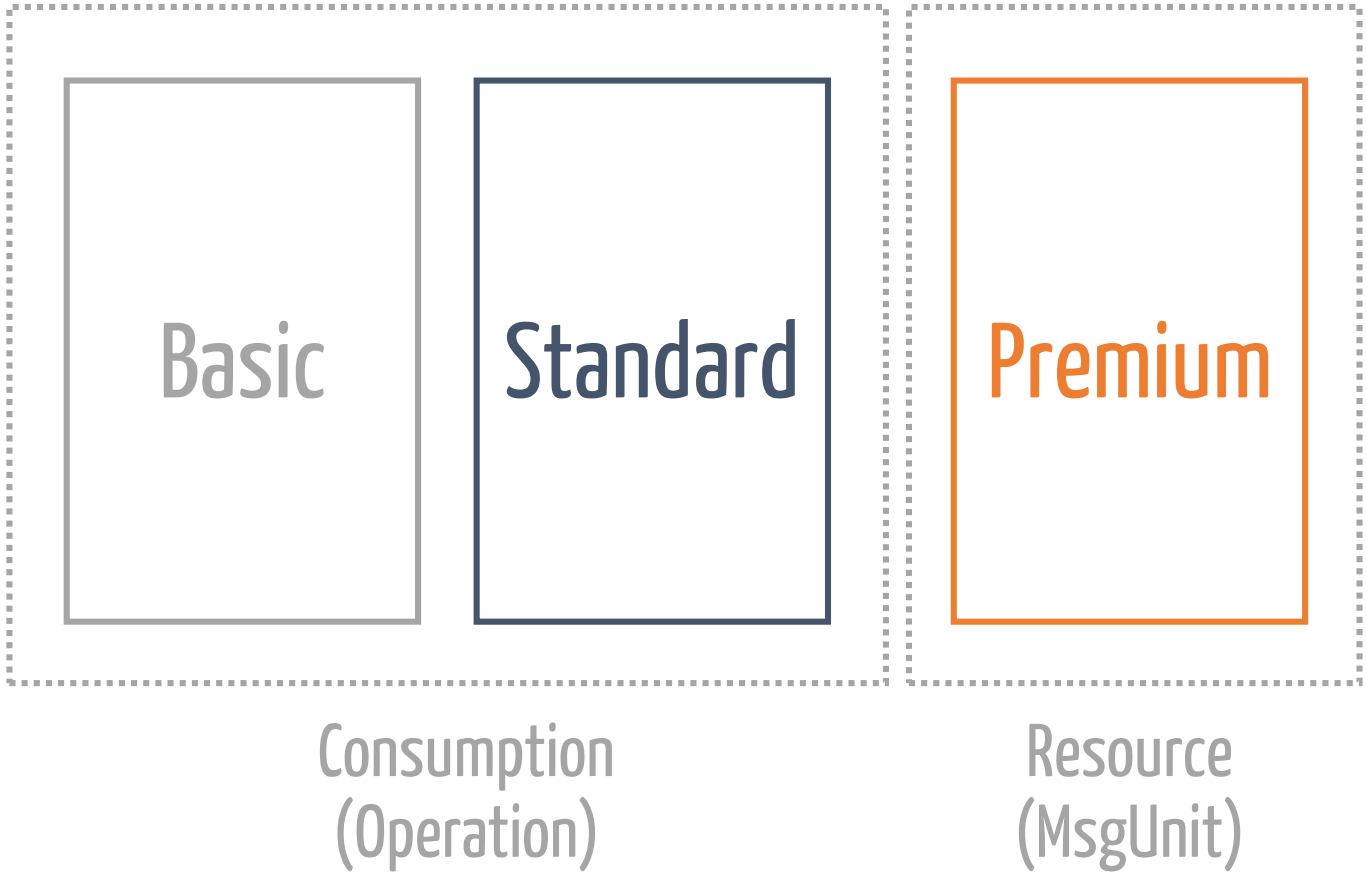
Storage Queues

Simple task queues

Service Bus

Enterprise messaging

Tiers


Basic
Standard
Premium
Consumption
(Operation)
Resource
(MsgUnit)


nuget

nuget Packages Upload Statistics Documentation Downloads Blog Sign in

azure servicebus

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/

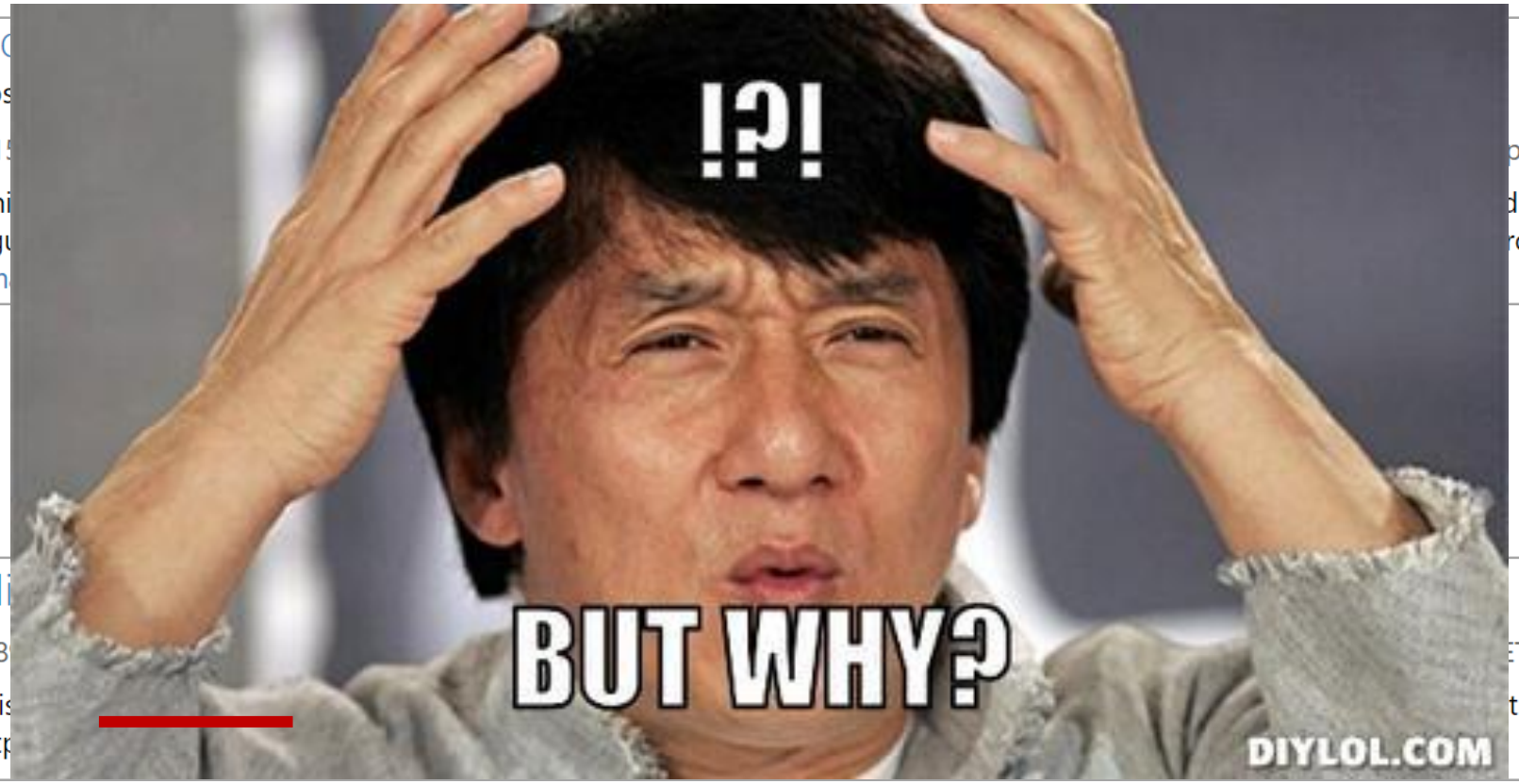
1

Wind

Micros

↓ 12,15

Use thi
configu
inform



pFabric Messaging PubSub P...

dll along with related
rosoft Azure... More

2

Mi

↓ 8

This
http

ET AMQP IoT Queue Topic

tion about Service Bus, see

Microsoft.Azure.ServiceBus

.Net Standard

.NET Framework

.NET Core

WindowsAzure.ServiceBus

nuget Packages Upload Statistics Documentation Downloads Blog Sign in

azure servicebus

3,600 packages returned for azure servicebus

☑ Include prerelease

Microsoft.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

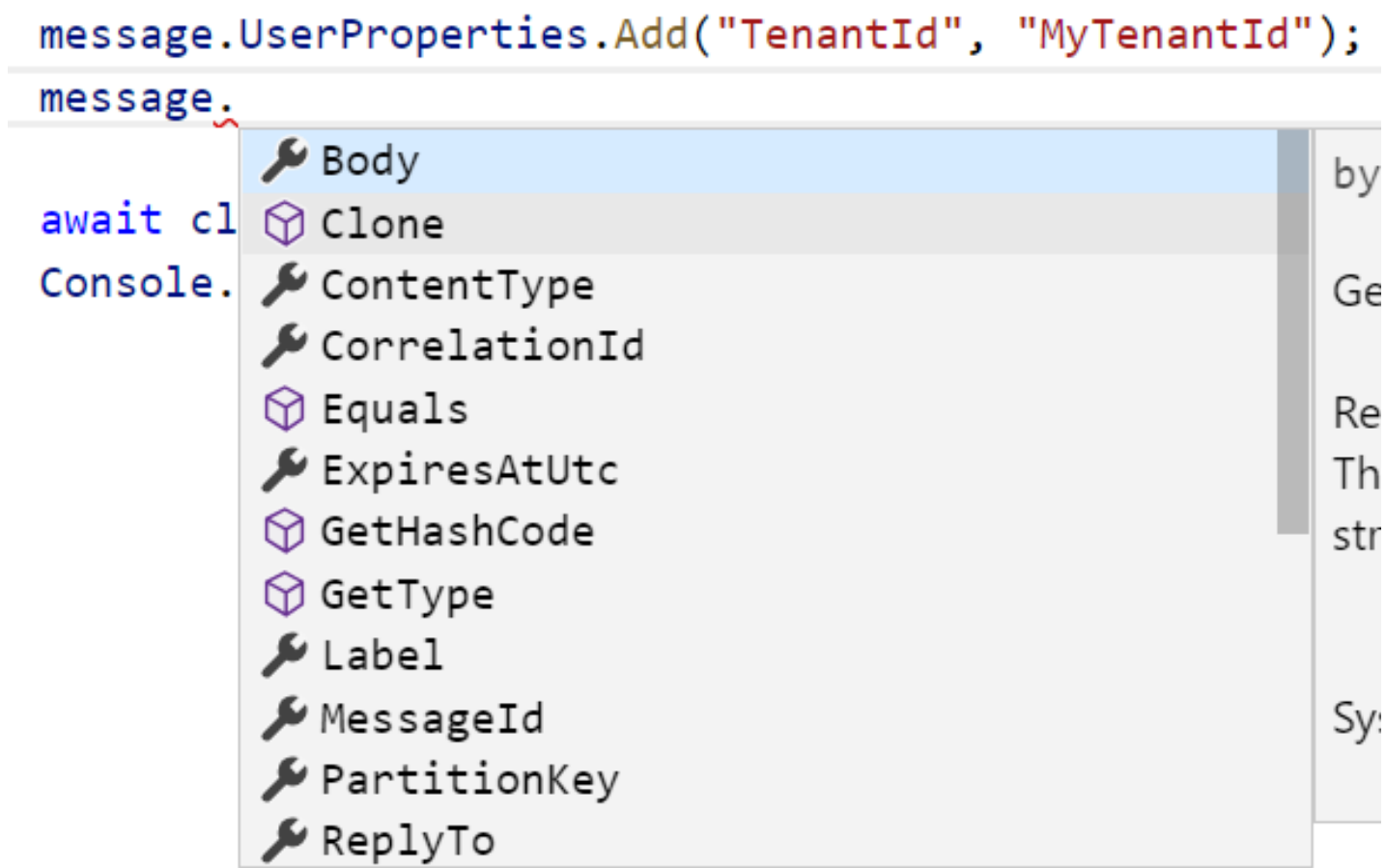
https://octodex.github.com/octobiwan

AMQP 1.0

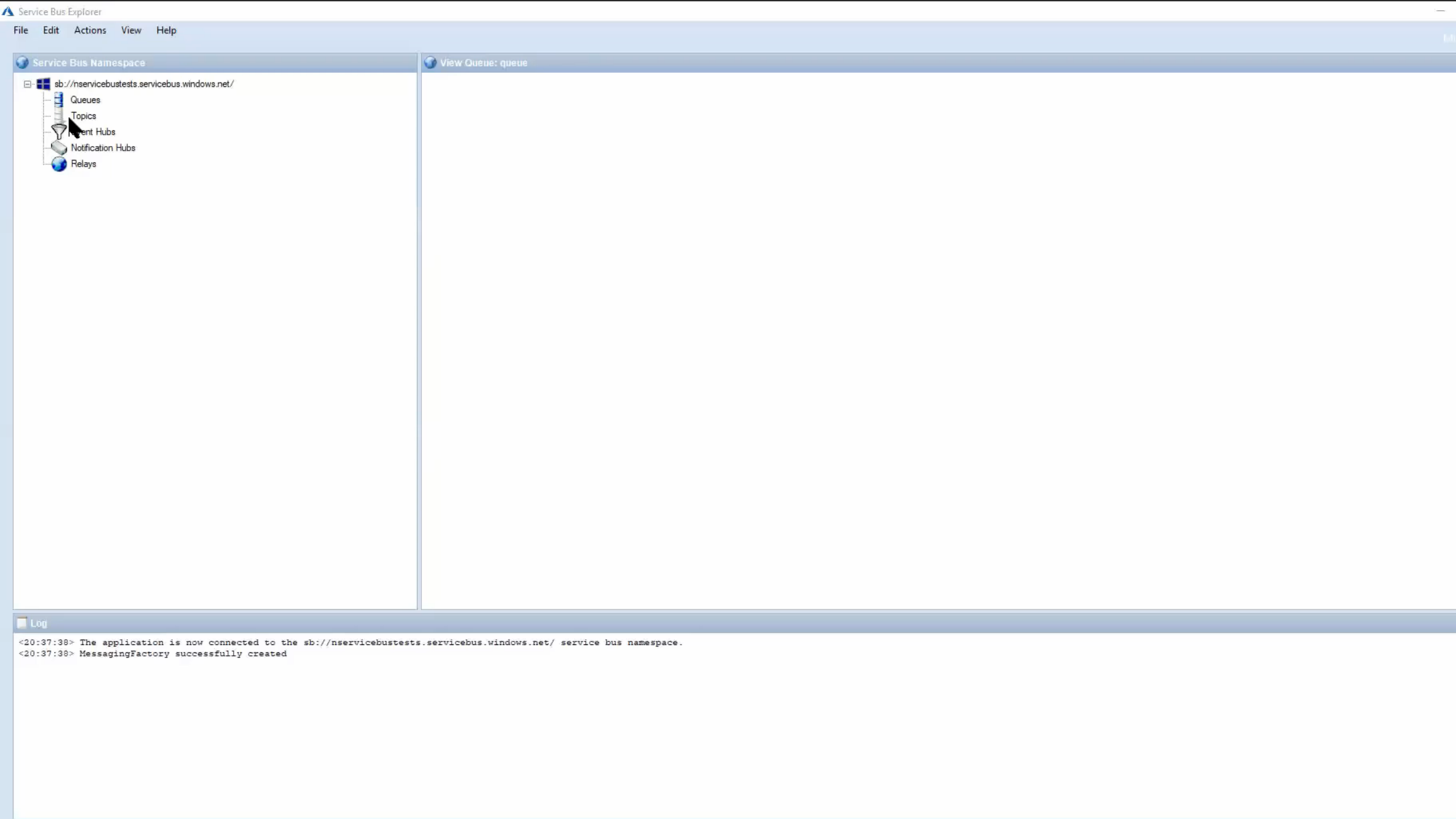
Send

```
<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
Clone
ContentType
CorrelationId
Equals
ExpiresAtUtc
GetHashCode
GetType
Label
MessageId
PartitionKey
ReplyTo

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


Service Bus Explorer
File
Edit
Actions
View
Help
Service Bus Namespace
sb://nservicebustests.servicebus.windows.net/
Queues
Topics
Event Hubs
Notification Hubs
Relays
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 Explorer

File Edit Actions View Help

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

Duplicate Detection History Time Window

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

Queue Properties

Max Queue Size In GB: 1 GB

Max Delivery Count: 10

User Description:

Forward To:

Forward Dead Lettered Messages To:

Auto Delete On Idle

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

Default Message Time To Live

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

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

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 | Upda

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");
```

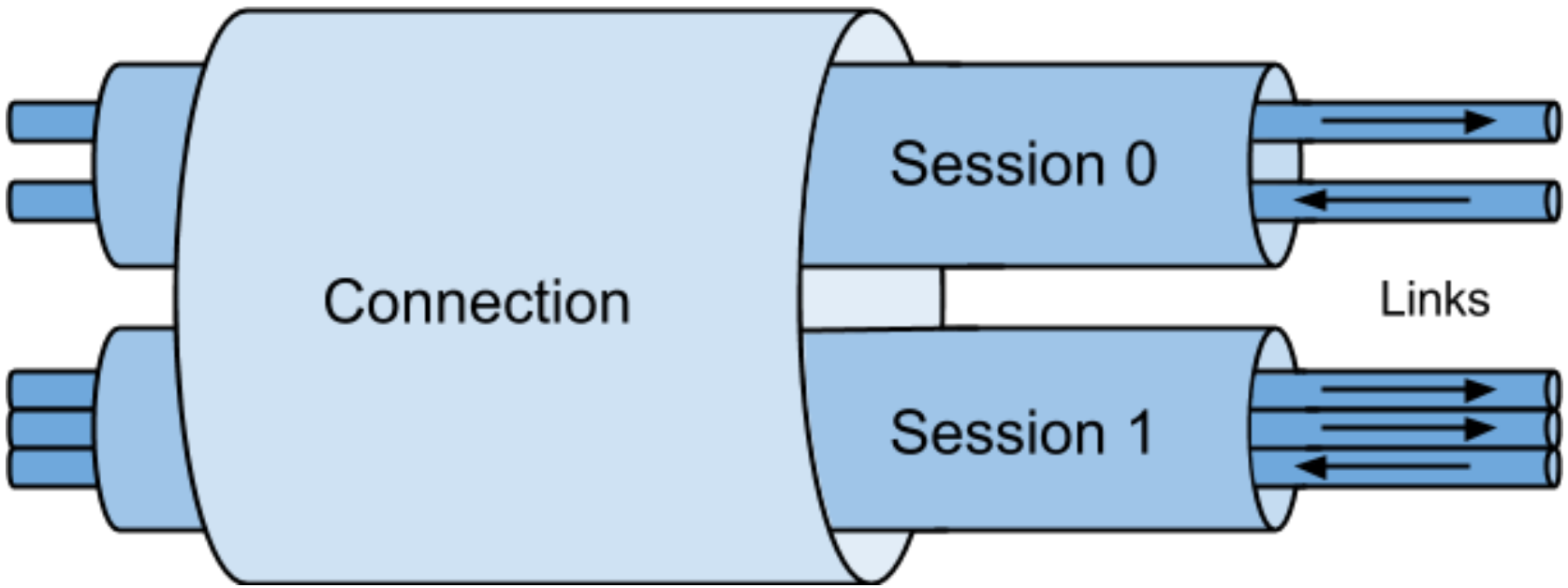
```
client.RegisterMessageHandler(
    async (message, token) =>
    {
        Console.WriteLine(
            $"Received message with '{message.MessageId}' and content '{Encoding.UTF8.GetString(message.Body)}'");
        // throw new InvalidOperationException();
        await client.CompleteAsync(message.SystemProperties.LockToken);
    },
    new MessageHandlerOptions(
        exception =>
        {
            Console.WriteLine($"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


Connection
Session 0
Session 1
Links

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");
```

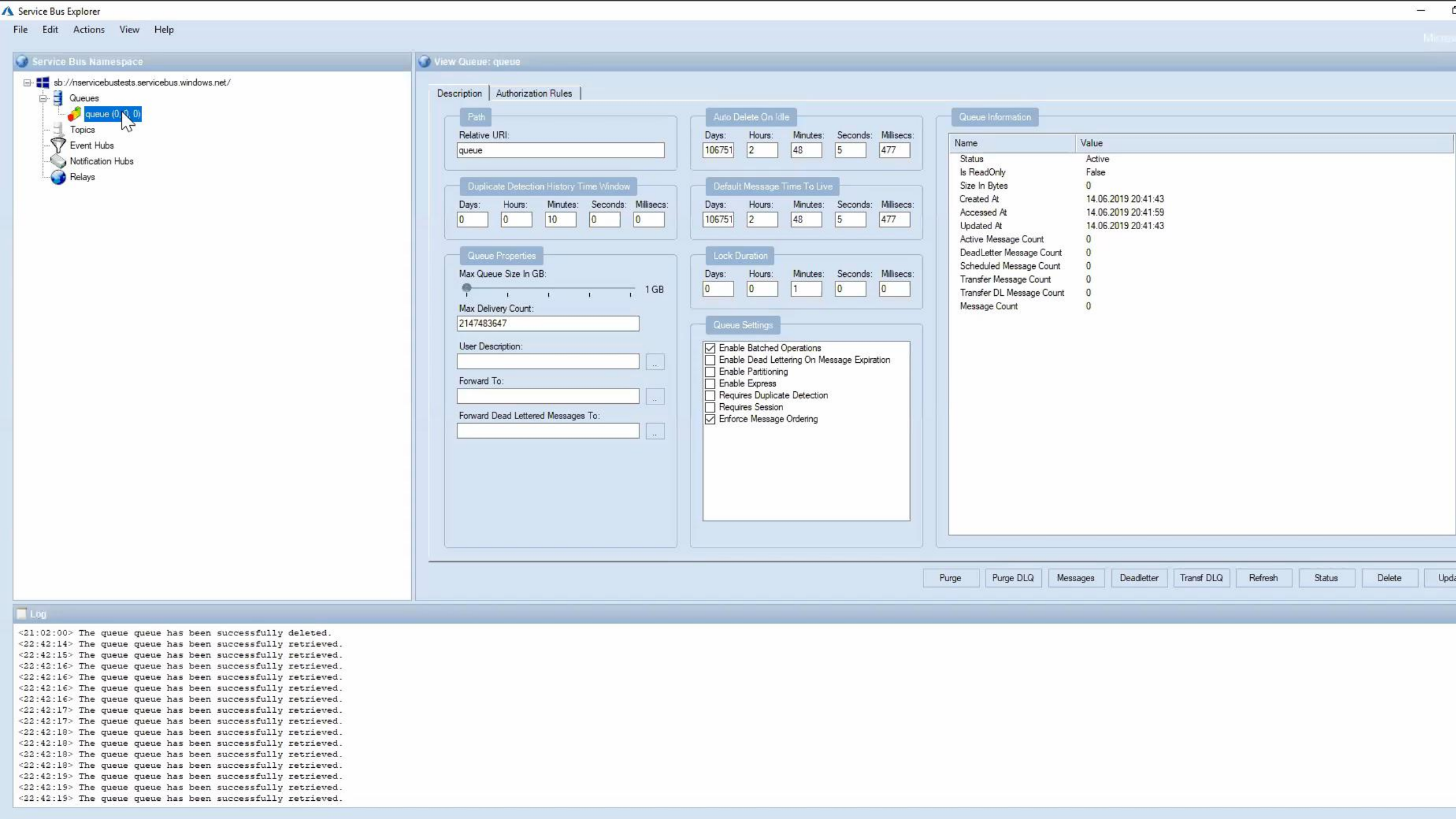
```
c:\p\AzureServiceBus.DeepDive\Scheduling
> dotnet run
```

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);



await Prepare.SimulateActiveReceiver(client);
```

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

Where does the message go?


Service Bus Explorer
File
Edit
Actions
View
Help
Service Bus Namespace
sb://nservicebustests.servicebus.windows.net/
Queues
queue (0, 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:
2147483647
User Description:
Forward To:
Forward Dead Lettered Messages To:
Lock Duration
Days: 0
Hours: 0
Minutes: 1
Seconds: 0
Millisecs: 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
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.
<22:42:17> 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.

Deadlettering

```
var description = new QueueDescription(destination)
{
    EnableDeadLetteringOnMessageExpiration = true, // default false
    MaxDeliveryCount = 1
};
await client.CreateQueueAsync(description);
```

```
var message = new Message();
message.Body = Encoding.UTF8.GetBytes("Half life");
message.TimeToLive = TimeSpan.FromSeconds(1);
await client.SendAsync(message);
```

```
client.RegisterMessageHandler(
    async (msg, token) =>
    {
        switch (Encoding.UTF8.GetString(msg.Body))
        {
            case "Half life":
                await client.AbandonAsync(msg.SystemProperties.LockToken);
                break;
        }
    },
```

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

Service Bus Explorer

File Edit Actions View Help

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

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 | Upd

Log

```
<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.
<22:44:42> 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.
```

Service Bus Explorer

File Edit Actions View Help

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
e11399faea7445249e520e...	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 Half life
```

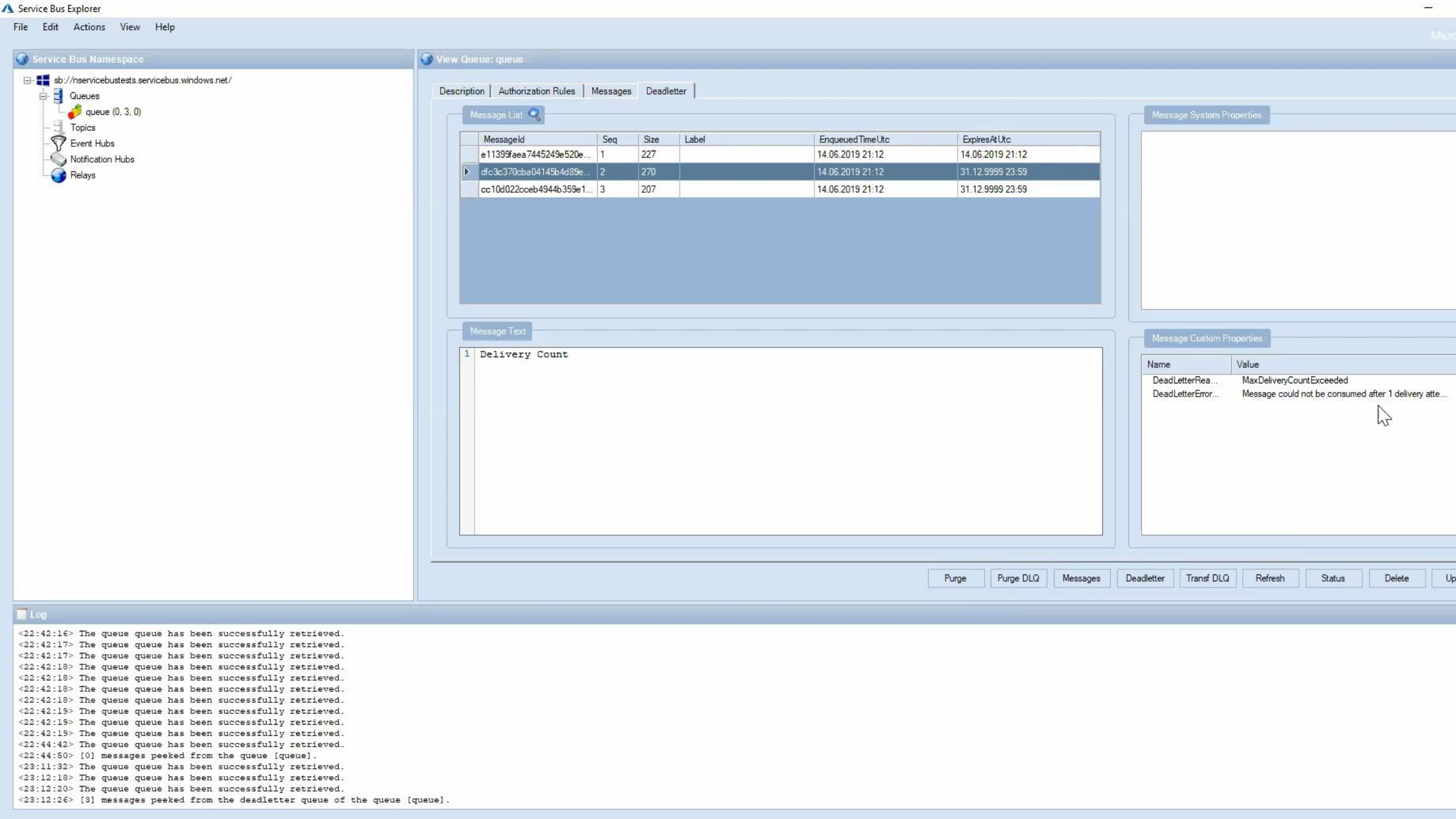
Message Custom Properties

Name	Value
DeadLetterRea...	TTLExpiredException
DeadLetterError...	The message expired and was dead lettered.

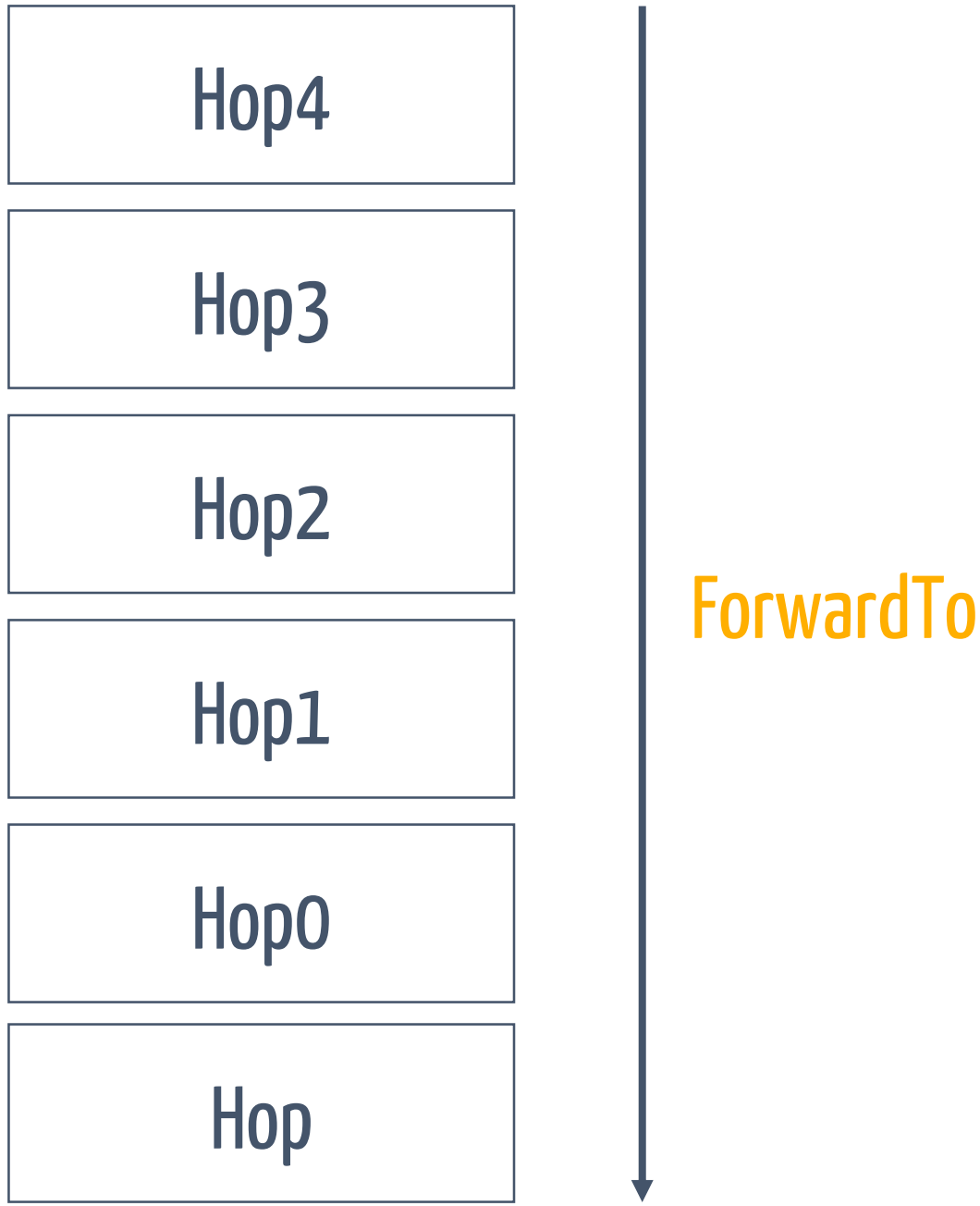
Purge | Purge DLQ | Messages | Deadletter | Transf DLQ | Refresh | Status | Delete | Up

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.
<22:42:18> 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.
<22:44:42> 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
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
e11399faea7445249e520e...
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
DeadLetterRea...
MaxDeliveryCountExceeded
DeadLetterError...
Message could not be consumed after 1 delivery atte...
Purge
Purge DLQ
Messages
Deadletter
Transf DLQ
Refresh
Status
Delete
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.
<22:42:18> 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.
<22:44:42> 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


Hop4
Hop3
Hop2
Hop1
Hop0
Hop
ForwardTo

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

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

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

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

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

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

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

File Edit Actions View Help

Service Bus Namespace

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

Entity

Log

File Edit Actions View Help

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

Duplicate Detection History Time Window

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

Queue Properties

Max Queue Size In GB: 1 GB

Max Delivery Count: 10

User Description:

Forward To:

Forward Dead Lettered Messages To:

Auto Delete On Idle

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

Default Message Time To Live

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

Lock Duration

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

Queue Settings

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

Queue Information

Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	0
Created At	14.06.2019 21:33:14
Accessed At	14.06.2019 21:33:20
Updated At	14.06.2019 21:33:14
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 | Del

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 'Weeeeeeehhh!' on hop 'Hop0'
Setup forwarding from Hop0 to Hop
```

File Edit Actions View Help

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 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

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

Queue Information

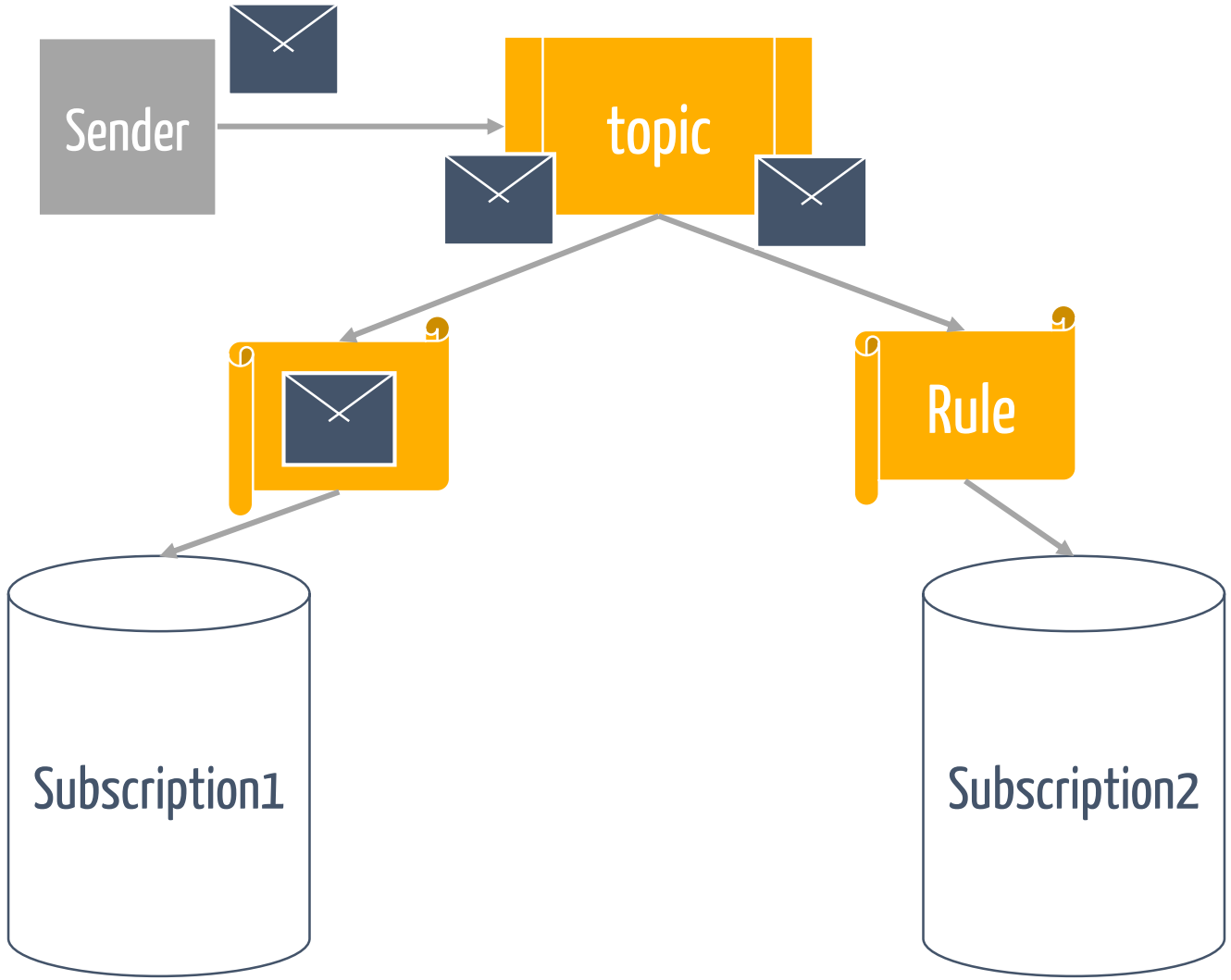
Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	0
Created At	14.06.2019 21:33:14
Accessed At	14.06.2019 21:33:20
Updated At	14.06.2019 21:33:14
Active Message Count	0
DeadLetter Message Count	0
Scheduled Message Count	0
Transfer Message Count	0
Transfer DL Message Count	0
Message Count	0

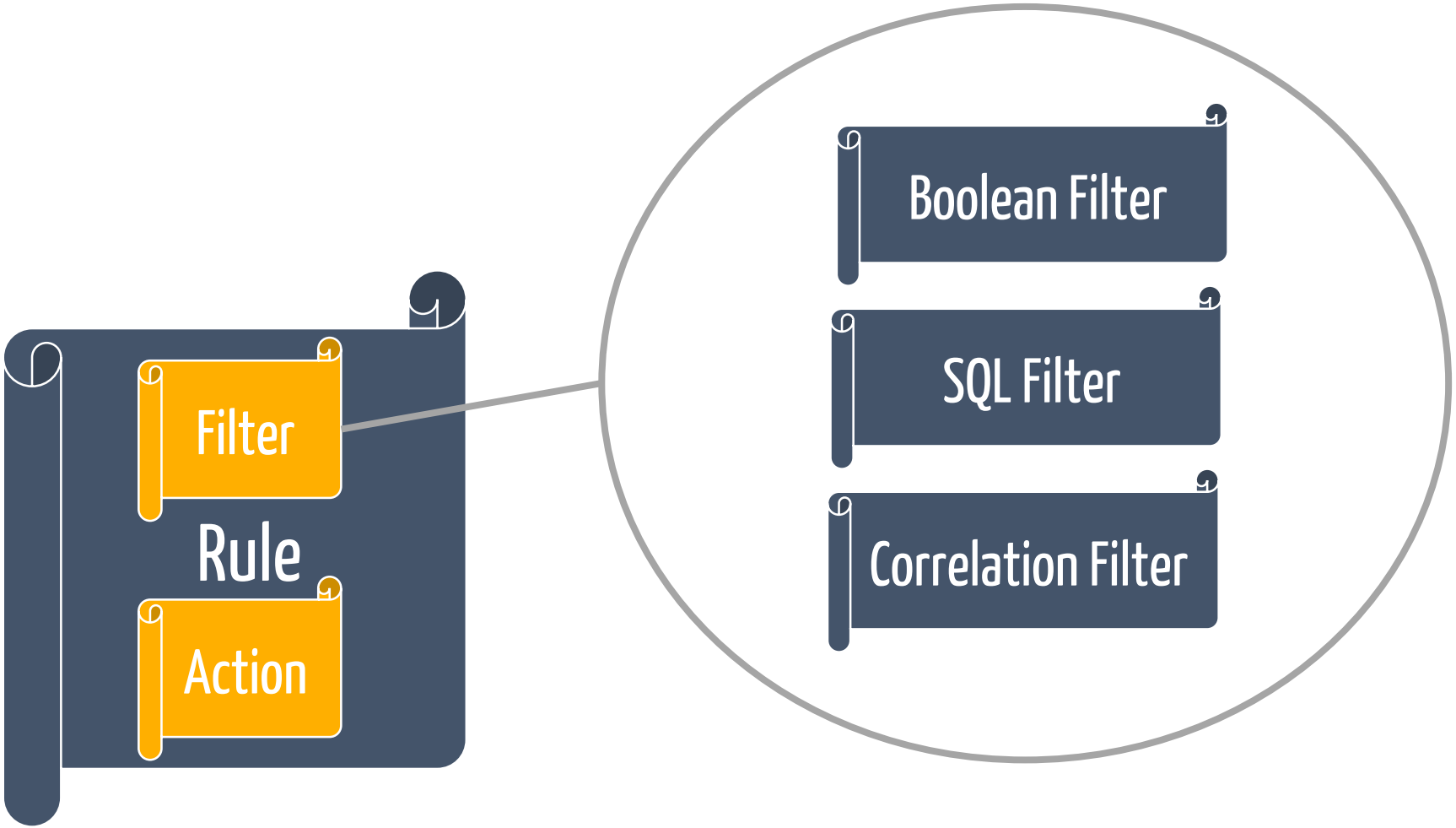
Purge | Purge DLQ | Messages | Deadletter | Transf DLQ | Refresh | Status | Delete

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.
```

Pub/Sub


Sender
topic
Rule
Subscription1
Subscription2


Filter
Rule
Action
Boolean Filter
SQL Filter
Correlation Filter

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

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

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

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

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

```
var message = new Message();
message.Body = Encoding.UTF8.GetBytes("Damn I have not time!");
message.Label = "rush";
await client.SendAsync(message);

message = new Message();
message.Body = Encoding.UTF8.GetBytes("I'm rich! I have 1000");
message.UserProperties.Add("currency", "CHF");
await client.SendAsync(message);
```

Service Bus Explorer

File Edit Actions View Help

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 Topic: topic

Description | Authorization Rules

Path

Relative URI: topic

Auto Delete On Idle

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

Default Message Time To Live

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

Topic Properties

Max Queue Size In GB: 1 GB

User Description:

Duplicate Detection History Time Window

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

Topic Settings

- [x] Enable Batched Operations
- [ ] Enable Filtering Messages Before Publishing
- [ ] Enable Partitioning
- [ ] Enable Express
- [ ] Requires Duplicate Detection
- [ ] Enforce Message Ordering

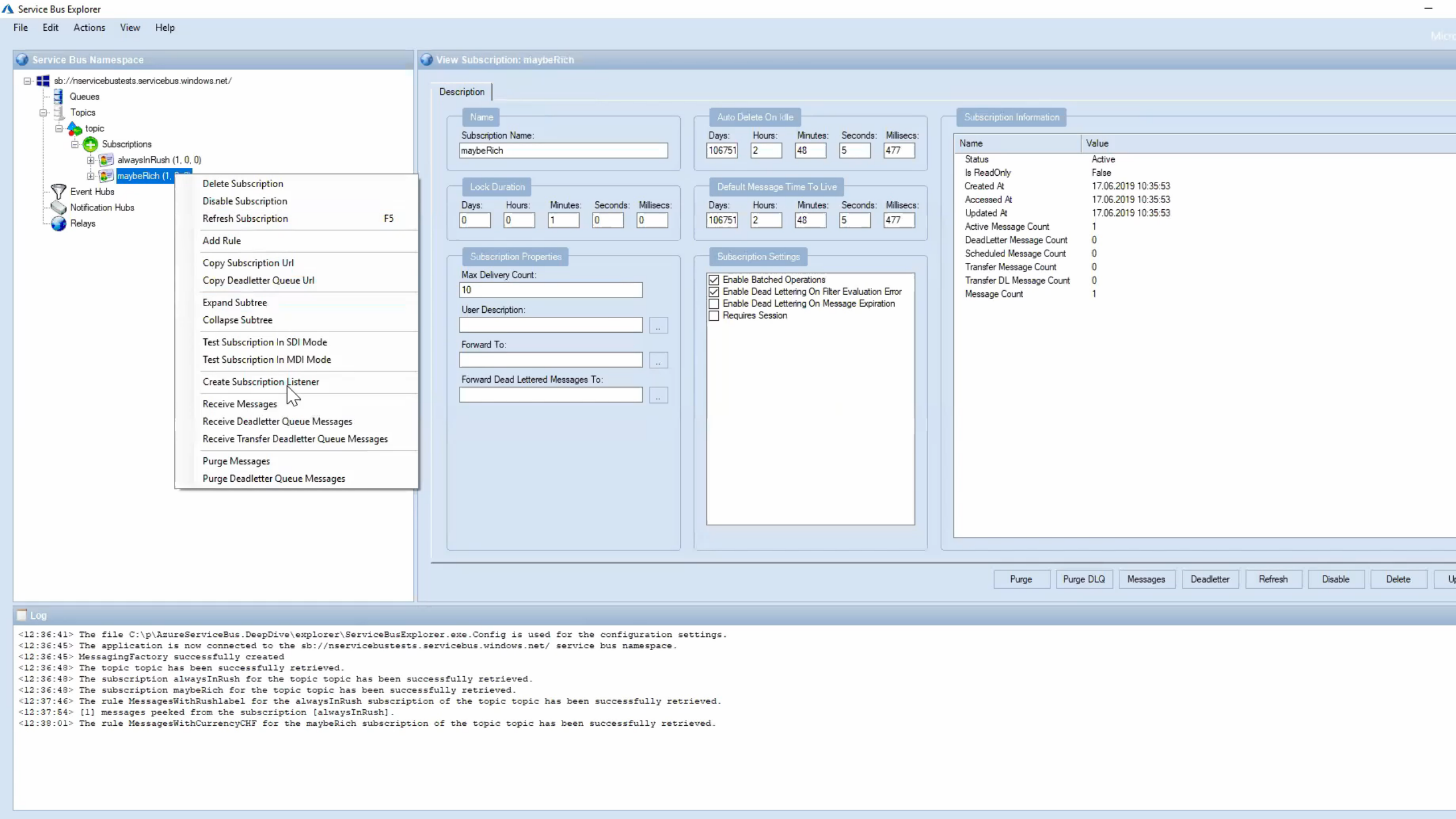
Topic Information

Name	Value
Status	Active
Is ReadOnly	False
Size In Bytes	504
Created At	17.06.2019 10:35:53
Accessed At	17.06.2019 10:35:55
Updated At	17.06.2019 10:35:53
Active Message Count	0
DeadLetter Message Count	0
Scheduled Message Count	0
Transfer Message Count	0
Transfer DL Message Count	0

Refresh Disable Delete

Log

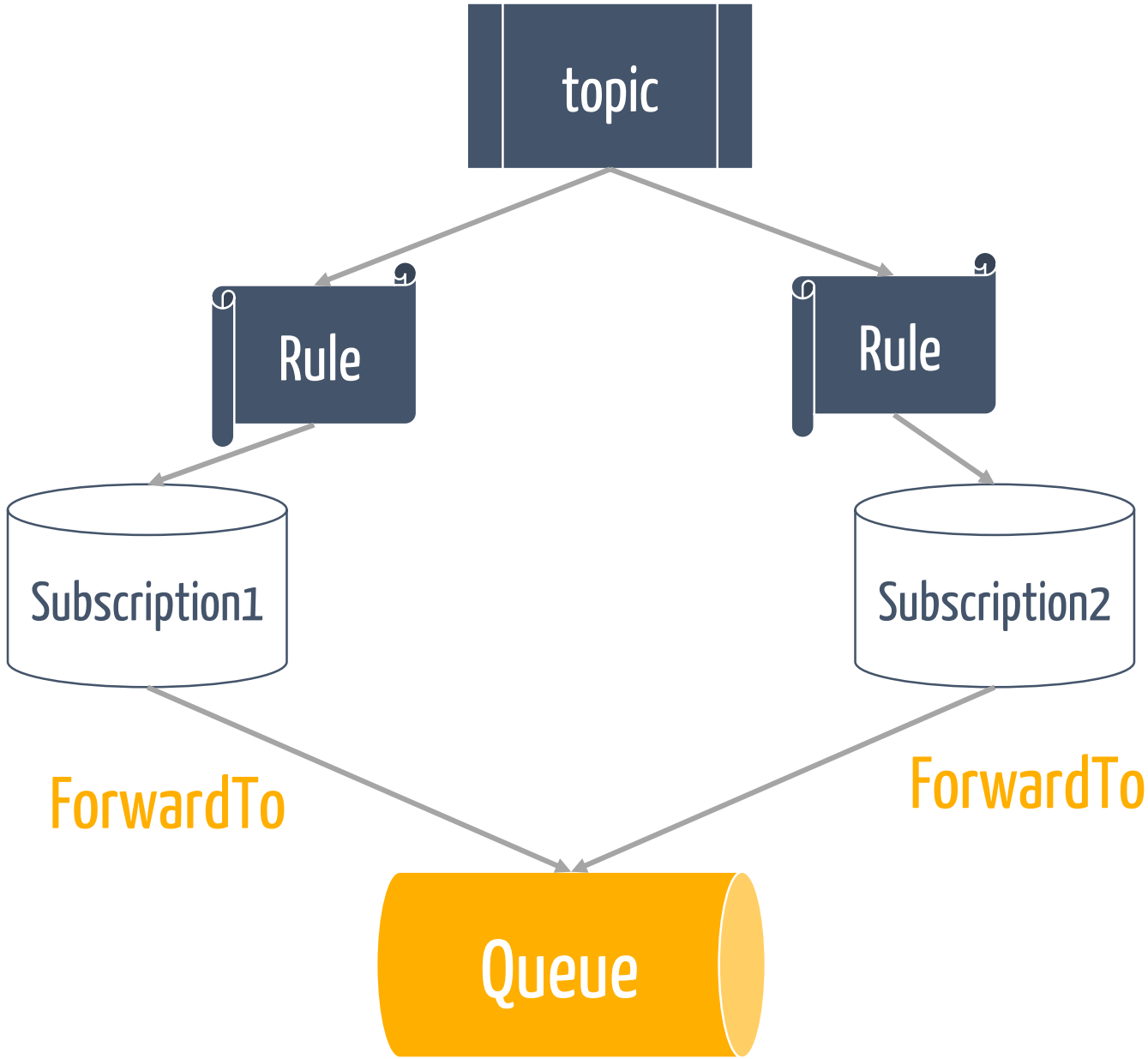
```
<12:36:41> The file C:\p\AzureServiceBus.DeepDive\explorer\ServiceBusExplorer.exe.Config is used for the configuration settings.
<12:36:45> The application is now connected to the sb://nservicebustests.servicebus.windows.net/ service bus namespace.
<12:36:45> MessagingFactory successfully created
<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.
```


Service Bus Explorer
File
Edit
Actions
View
Help
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
Delete Subscription
Disable Subscription
Refresh Subscription F5
Add Rule
Copy Subscription Url
Copy Deadletter Queue Url
Expand Subtree
Collapse Subtree
Test Subscription In SDI Mode
Test Subscription In MDI Mode
Create Subscription Listener
Receive Messages
Receive Deadletter Queue Messages
Receive Transfer Deadletter Queue Messages
Purge Messages
Purge Deadletter Queue Messages
View Subscription: maybeRich
Description
Name
Subscription Name:
maybeRich
Auto Delete On Idle
Days: Hours: Minutes: Seconds: Millisecs:
106751 2 48 5 477
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
Lock Duration
Days: Hours: Minutes: Seconds: Millisecs:
0 0 1 0 0
Default Message Time To Live
Days: Hours: Minutes: Seconds: Millisecs:
106751 2 48 5 477
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
Purge
Purge DLQ
Messages
Deadletter
Refresh
Disable
Delete
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


topic
Rule
Rule
Subscription1
Subscription2
ForwardTo
ForwardTo
Queue

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

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

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

Service Bus Explorer

File Edit Actions View Help

Service Bus Namespace

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

View Subscription: maybeRich

Description | Messages

Message List

MessageId	Seq	Size	Label	EnqueuedTimeUtc	ExpiresAtUtc
e0d9141d27684e3aac0eb6...	1	163		17.06.2019 10:35	31.12.9999 23:59

Message System Properties

Misc	
ContentType	
CorrelationId	
DeadLetterSource	
DeliveryCount	1
EnqueuedSequenceNumber	2
EnqueuedTimeUtc	17.06.2019 10:35
ExpiresAtUtc	31.12.9999 23:59
ForcePersistence	**False**
IsBodyConsumed	False
Label	
LockedUntilUtc	Operation is not valid due to the current s
LockToken	Operation is not valid due to the current s
MessageId	**e0d9141d27684e3aac0eb66ea0a**

Message Text

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

Message Custom Properties

Name	Value
currency	Zloty
RuleName	MessagesWithCurrencyCHF

Purge | Purge DLQ | Messages | Deadletter | Refresh | Disable | Delete

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.
```

Service Bus Explorer

File Edit Actions View Help

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	1
EnqueuedTimeUtc	17.06.2019 10:52
ExpiresAtUtc	31.12.9999 23:59
ForcePersistence	**False**
IsBodyConsumed	False
Label	**rush**
LockedUntilUtc	Operation is not valid due to the current s
LockToken	Operation is not valid due to the current s
MessageId	6b1fe11b26be418ab3b349f43c3f...

Message Text

```
1 Damn I have not time!
```

Message Custom Properties

Name	Value

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

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

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

using (var scope = new TransactionScope(TransactionScopeAsyncFlowOption.Enabled))
{
    var message = new Message(Encoding.UTF8.GetBytes("Deep Dive 1"));
    await client.SendAsync(message);

    message = new Message(Encoding.UTF8.GetBytes("Deep Dive 2"));
    await client.SendAsync(message);

    scope.Complete();
}
```

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

Batching

```
var messages = new List<Message>();
for (var i = 0; i < 10; i++)
{
    var message = new Message();
    message.Body = Encoding.UTF8.GetBytes($"Deep Dive{i}");
    messages.Add(message);
}

await client.SendAsync(messages);
```

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

Upgrade to premium tier

Use ServiceBus.AttachmentPlugin

Send Via


Incoming
Transfer
Outgoing1
Outgoing2

```
connection = new ServiceBusConnection(connectionString);
receiver = new MessageReceiver(connection, inputQueue);
sender = new MessageSender(connection, destinationQueue, inputQueue);
receiver.RegisterMessageHandler(
    async (message, token) =>
    {
        using (var scope = new TransactionScope(TransactionScopeAsyncFlowOption.Enabled))
        {
            await sender.SendAsync(new Message(Encoding.UTF8.GetBytes("Will not leak")));

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

            await sender.SendAsync(new Message(Encoding.UTF8.GetBytes("Will not leak")));

            scope.Complete();
        }

        await Prepare.ReportNumberOfMessages(connectionString, destinationQueue);
    },
    Prepare.Options(connectionString, destinationQueue)
);
```

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

```
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'
```

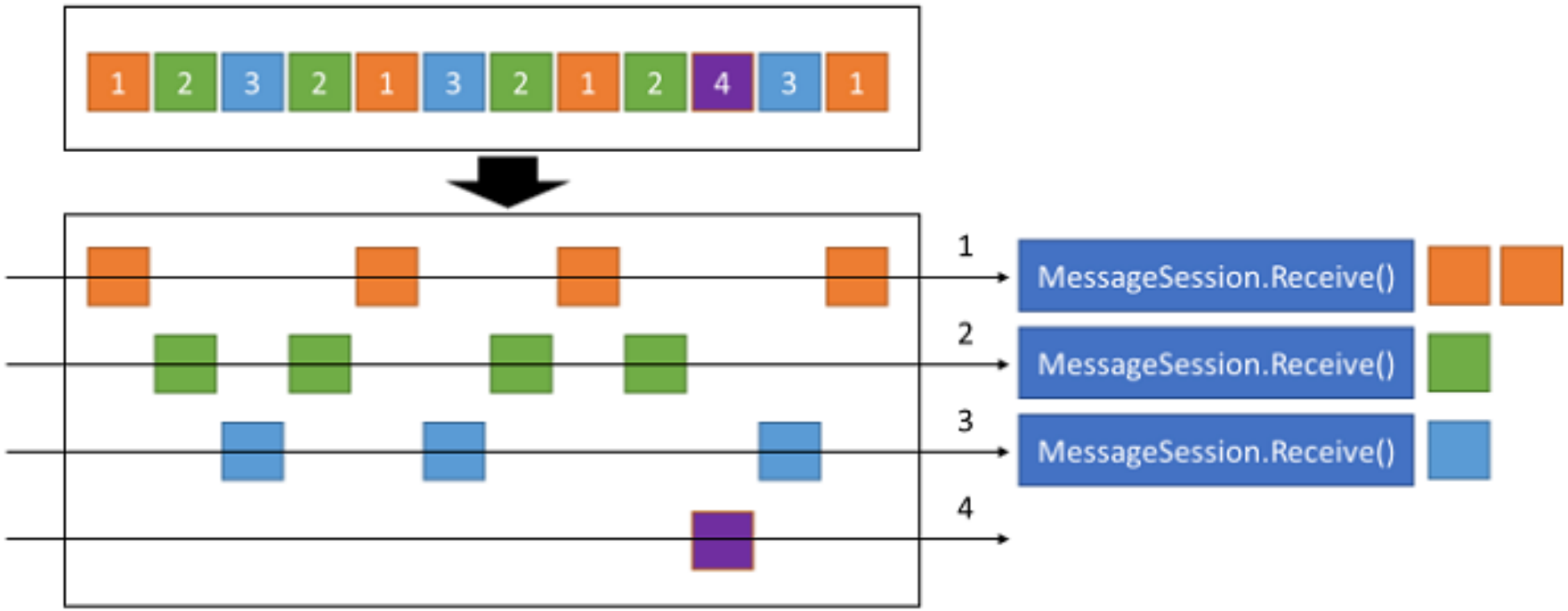
TransferDLQ

```
receiver.RegisterMessageHandler(
    async (message, token) =>
    {
        us                                                          .Enabled))
        {   var description = new QueueDescription(destinationQueue)
            {                                                                leak")));
                Status = EntityStatus.SendDisabled
            };
            await client.UpdateQueueAsync(description);
        }

        await Prepare.ReportNumberOfMessages(connectionString, inputQueue);
    }, Prepare.Options(connectionString, inputQueue)
);
```

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

var info = await client.GetQueueRuntimeInfoAsync(destination);

long activeMessageCount = info.MessageCountDetails.ActiveMessageCount;
long deadLetterMessageCount = info.MessageCountDetails.DeadLetterMessageCount;
long transferDeadLetterMessageCount = info.MessageCountDetails.TransferDeadLetterMessageCount;

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

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

Sessions


1 2 3 2 1 3 2 1 2 4 3 1
1
2
3
4
MessageSession.Receive()
MessageSession.Receive()
MessageSession.Receive()

```
var queueDescription = new QueueDescription(destination)
{
    RequiresSession = true
};
await client.CreateQueueAsync(queueDescription);
```

```
var messages = new List<Message>
{
    new Message(Encoding.UTF8.GetBytes("Orange 1")) {SessionId = "Orange"},
    new Message(Encoding.UTF8.GetBytes("Green 1")) {SessionId = "Green"},
    new Message(Encoding.UTF8.GetBytes("Blue 1")) {SessionId = "Blue"},
    new Message(Encoding.UTF8.GetBytes("Green 2")) {SessionId = "Green"},
    new Message(Encoding.UTF8.GetBytes("Orange 2")) {SessionId = "Orange"},
    new Message(Encoding.UTF8.GetBytes("Blue 2")) {SessionId = "Blue"},
    new Message(Encoding.UTF8.GetBytes("Green 3")) {SessionId = "Green"},
    new Message(Encoding.UTF8.GetBytes("Orange 3")) {SessionId = "Orange"},
    new Message(Encoding.UTF8.GetBytes("Green 4")) {SessionId = "Green"},
    new Message(Encoding.UTF8.GetBytes("Purple 1")) {SessionId = "Purple"},
    new Message(Encoding.UTF8.GetBytes("Blue 3")) {SessionId = "Blue"},
    new Message(Encoding.UTF8.GetBytes("Orange 4")) {SessionId = "Orange"}
};

await client.SendAsync(messages);
```

```
client.RegisterSessionHandler(
    (session, message, token) =>
    {
        Console.WriteLine(
            $"Received message on session '{session.SessionId}' with '{message.MessageId}' and content '
            {Encoding.UTF8.GetString(message.Body)}'");
        return Task.CompletedTask;
    },
    new SessionHandlerOptions(
        exception => ···
    {
        AutoComplete = true,
        MaxConcurrentSessions = 1,
        MessageWaitTimeout = TimeSpan.FromSeconds(2),
        MaxAutoRenewDuration = TimeSpan.FromMinutes(10)
    }
);
```

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

Dedup

```
var queueDescription = new QueueDescription(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<Message>
{
    new Message(content) { MessageId = messageId },
    new Message(content) { MessageId = messageId },
    new Message(content) { MessageId = messageId }
};

await client.SendAsync(messages);
```

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

Plugins

```
class MyCustomPlugin : ServiceBusPlugin
{
    public override string Name => "MyCustomPlugin";

    public override Task<Message> BeforeMessageSend(Message message)
    {
        var currentBody = Encoding.UTF8.GetString(message.Body);

        var client = new QueueClient(connectionString, destination);
        client.RegisterPlugin(new MyCustomPlugin());

    public override Task<Message> AfterMessageReceive(Message message)
    {
        var currentBody = Encoding.UTF8.GetString(message.Body);
        message.Body = Encoding.UTF8.GetBytes($"{currentBody}{Environment.NewLine}Yes I can before receive");
        return Task.FromResult(message);
    }
}
```

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

SUMMARY

ENTERPRISE MESSAGING FEATURES

MESSAGE TRANSACTIONAL PROCESSING

RELIABILITY

GUARANTEED THROUGHPUT AND LATENCY

TOTAL: PRICELESS

Particular
Software

Thanks

Slides, Links...

github.com/danielmarbach/AzureServiceBus.DeepDive

Particular
Software

Q&A

Particular Software

Software Engineer
Microsoft MVP

@danielmarbach
particular.net/blog
planetgeek.ch

Particular
Software

Thanks