Introducing RabbitMQ Exchanges

Stephen Haunts
DEVELOPER, LEADER, AUTHOR AND TRAINER

@stephenhaunts   www.stephenhaunts.com

Overview

AMQP Protocol

Exchanges

- Direct exchange
- Fanout exchange
- Topic exchange
- Headers exchange

Queues, bindings and consumers

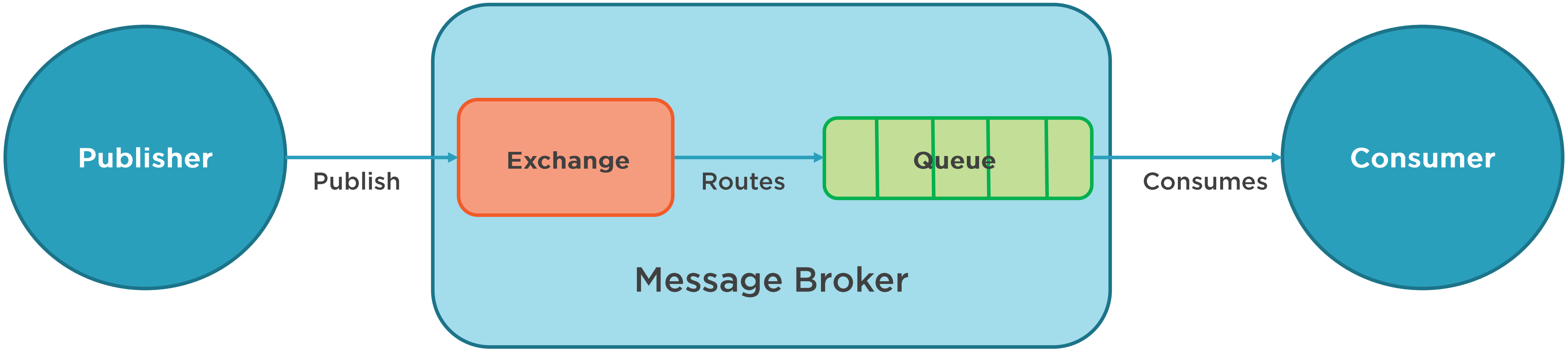
Code demonstration

AMQP Messaging Standard

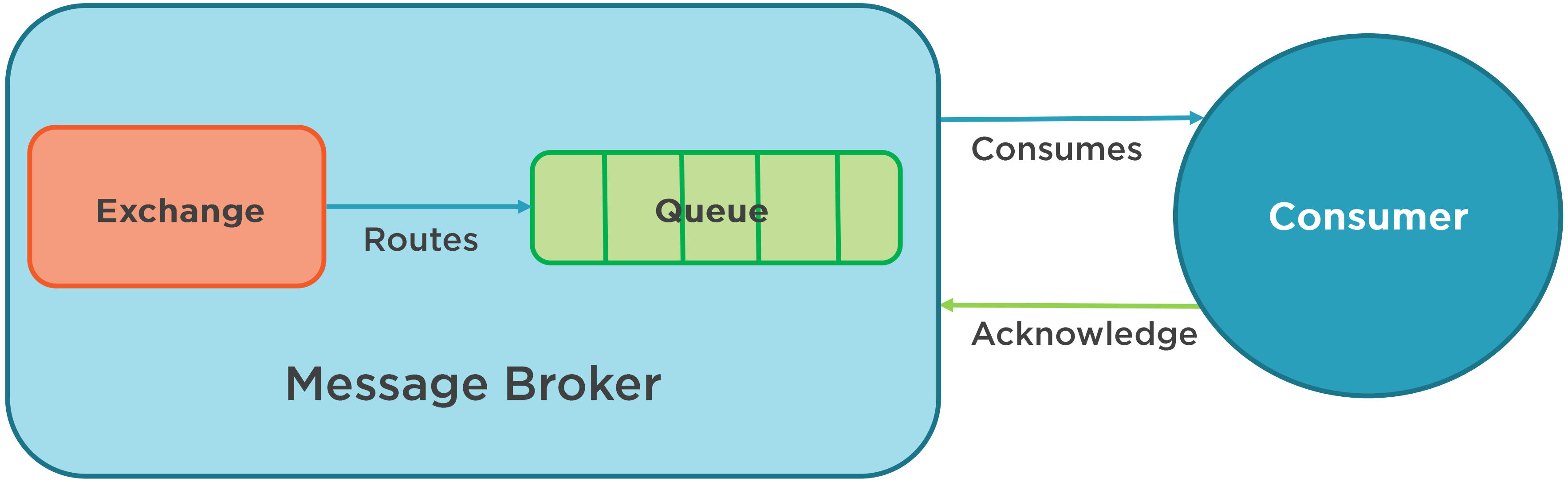
Advanced Message Queueing Protocol

RabbitMQ supports version 0-9-1

AMQP Messaging Standard



AMQP Messaging Standard



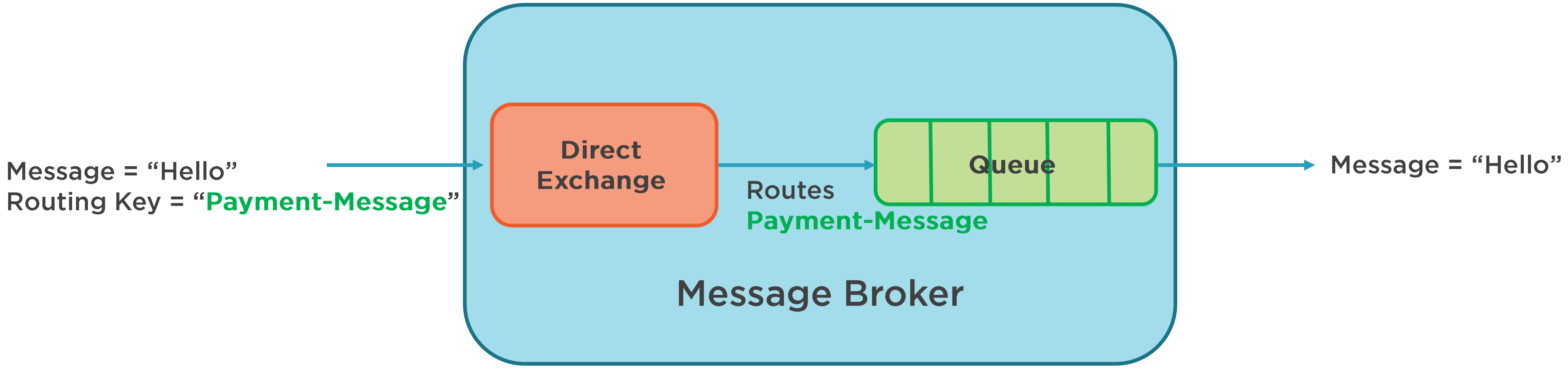
Exchanges

Direct Exchanges	Fanout Exchanges
Topic Exchanges	Header Exchanges

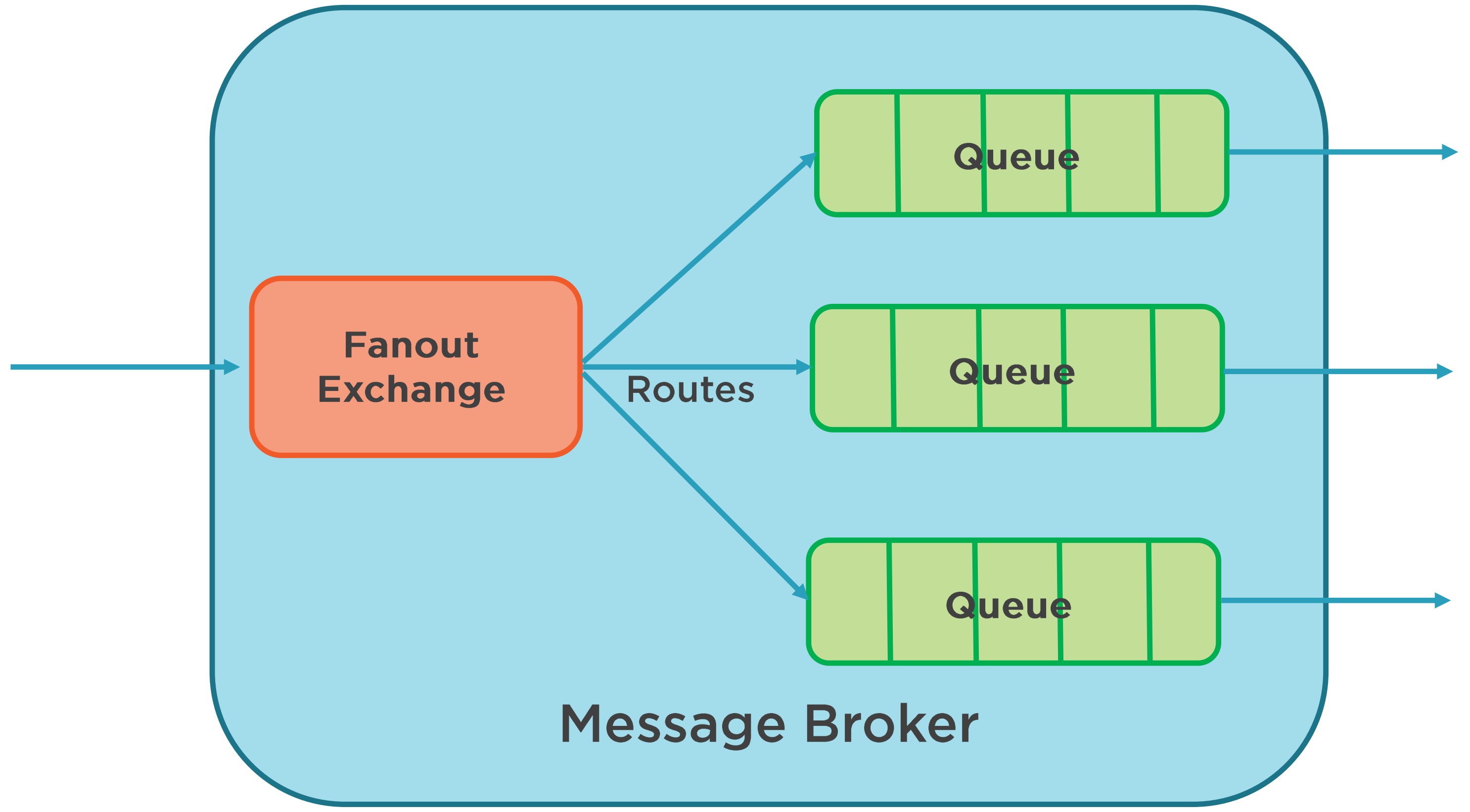
Exchanges

Name	*The name of the exchange*
Durability	*Persisting the messages to disk*
Auto-Delete	*Delete message when not needed*
Arguments	*These are message broker-dependent*

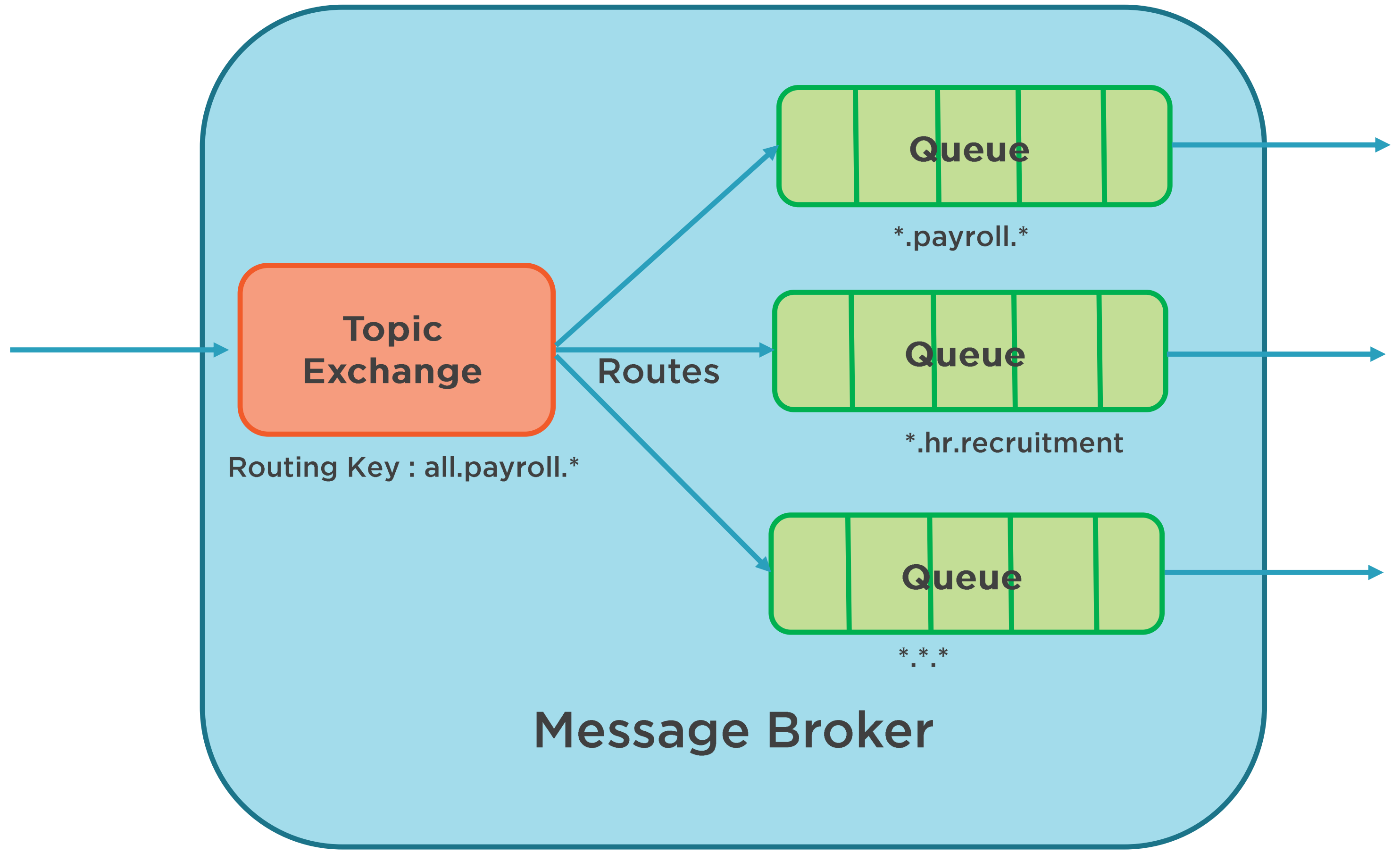
Direct Exchange



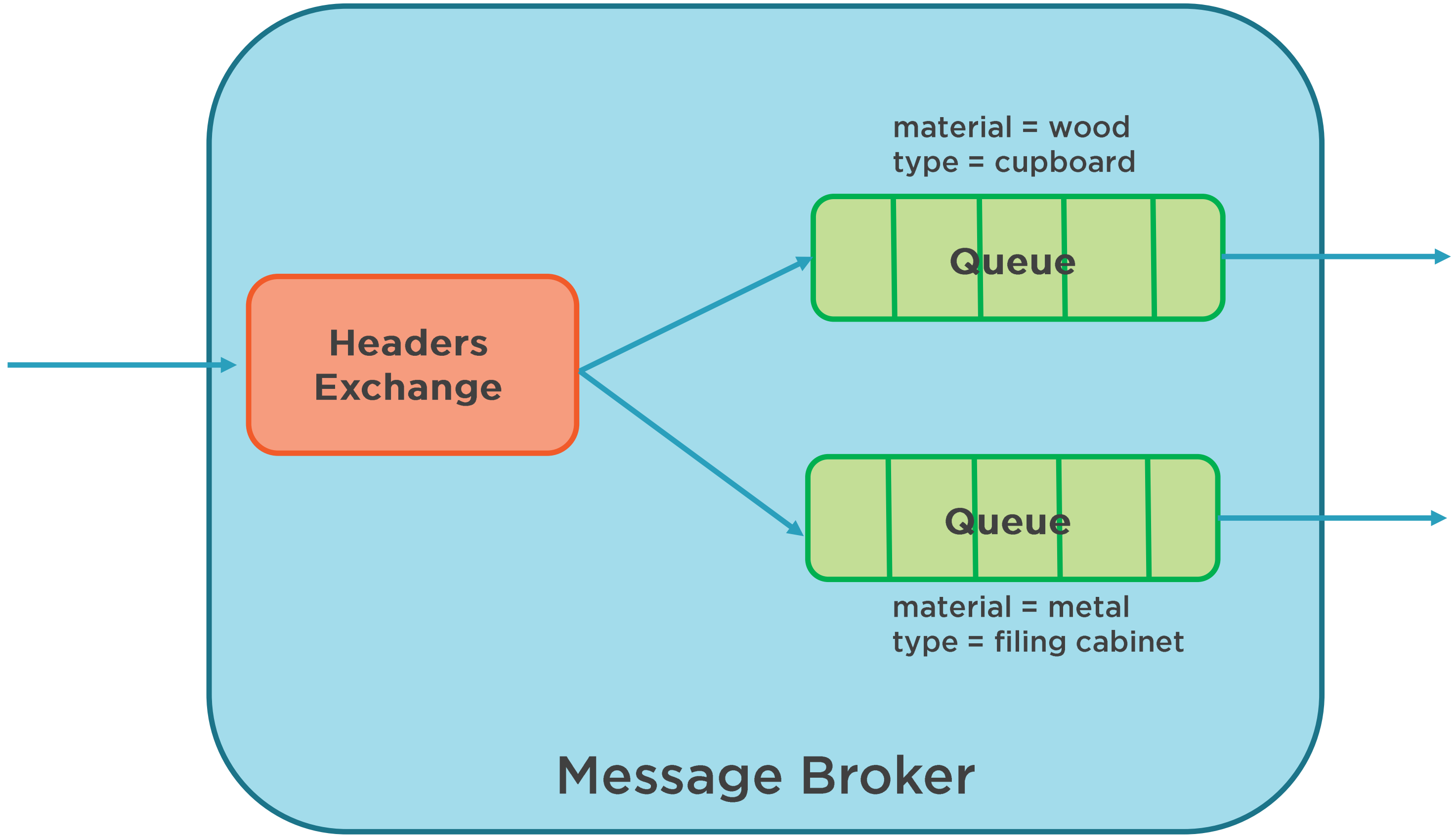
Fanout Exchange



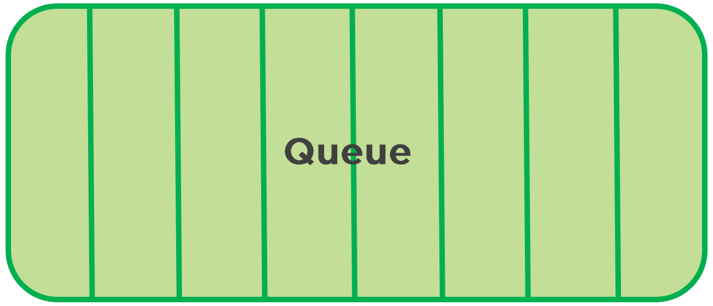
Topic Exchange



Headers Exchange

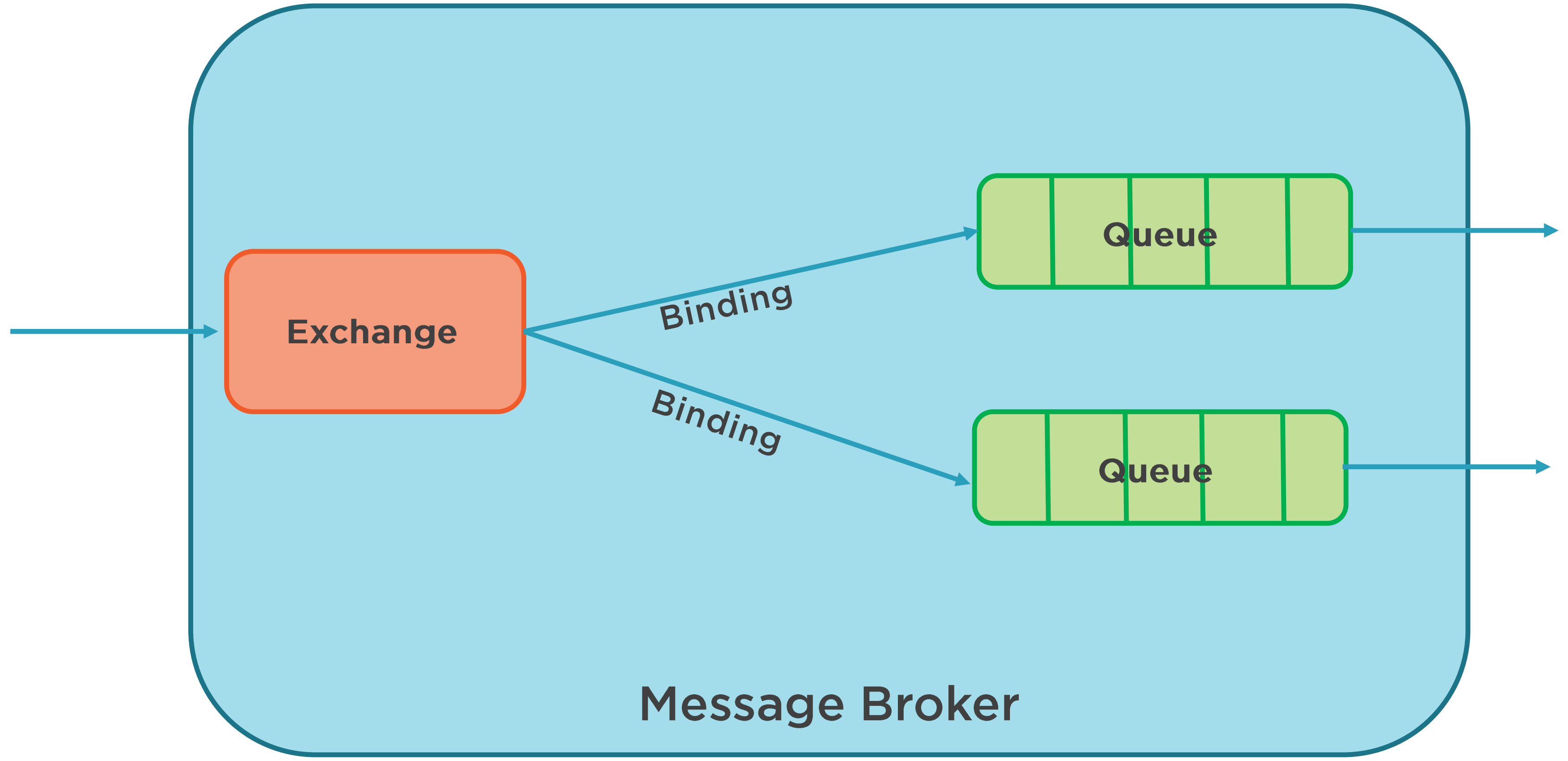


Queues

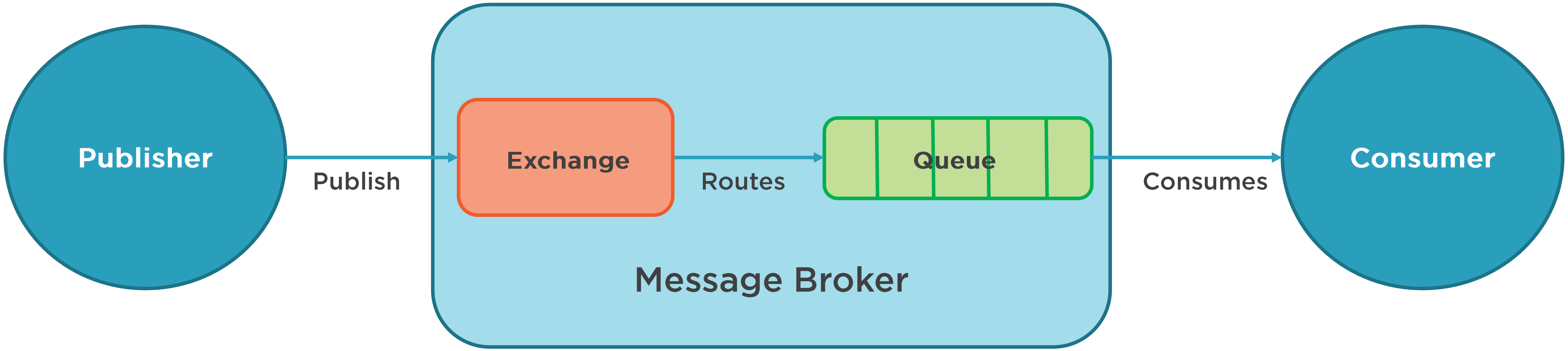


Name	*The name of the queue*
Durable	*Persisting the queue to disk*
Exclusive	*Delete queue when not needed*
Auto Delete	*Queue deleted when consumer unsubscribes*

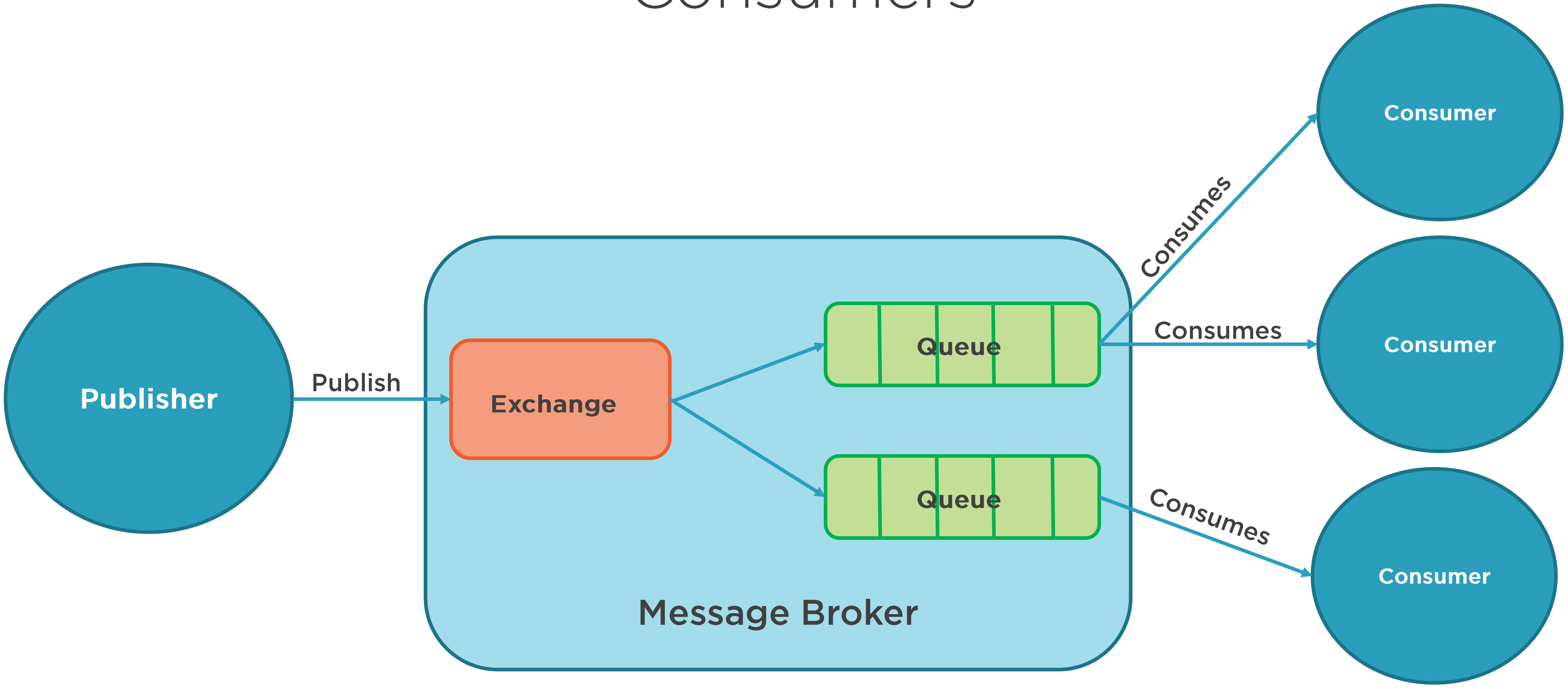
Bindings



Consumers

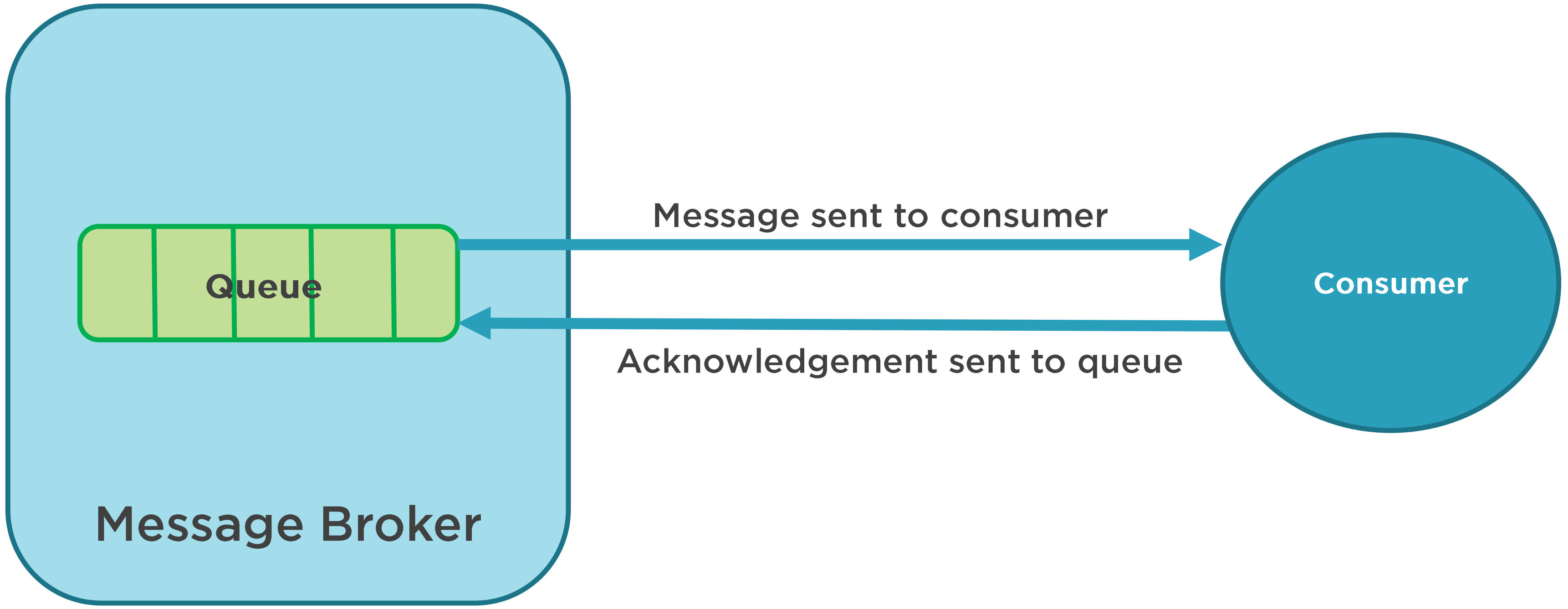

Publisher
Publish
Exchange
Routes
Queue
Consumes
Consumer
Message Broker

Consumers


Publisher
Publish
Exchange
Queue
Queue
Message Broker
Consumes
Consumes
Consumes
Consumer
Consumer
Consumer

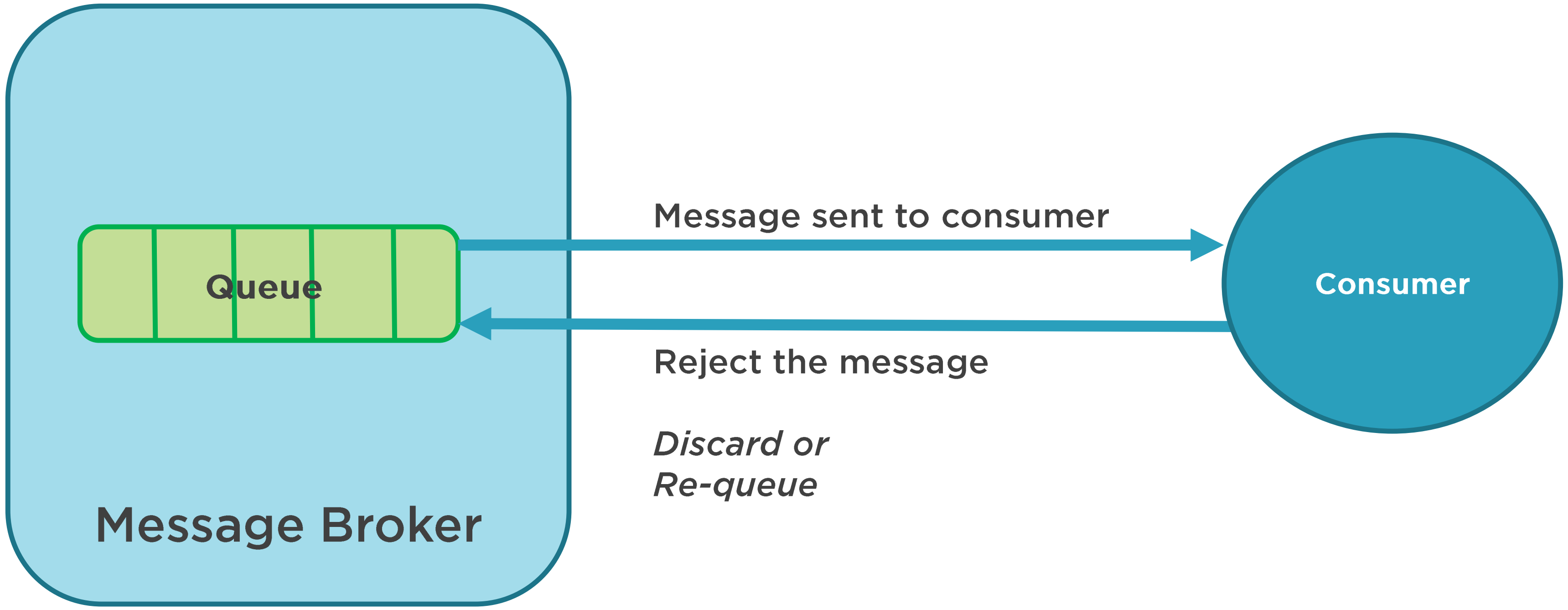
Consumers

Message Acknowledgements

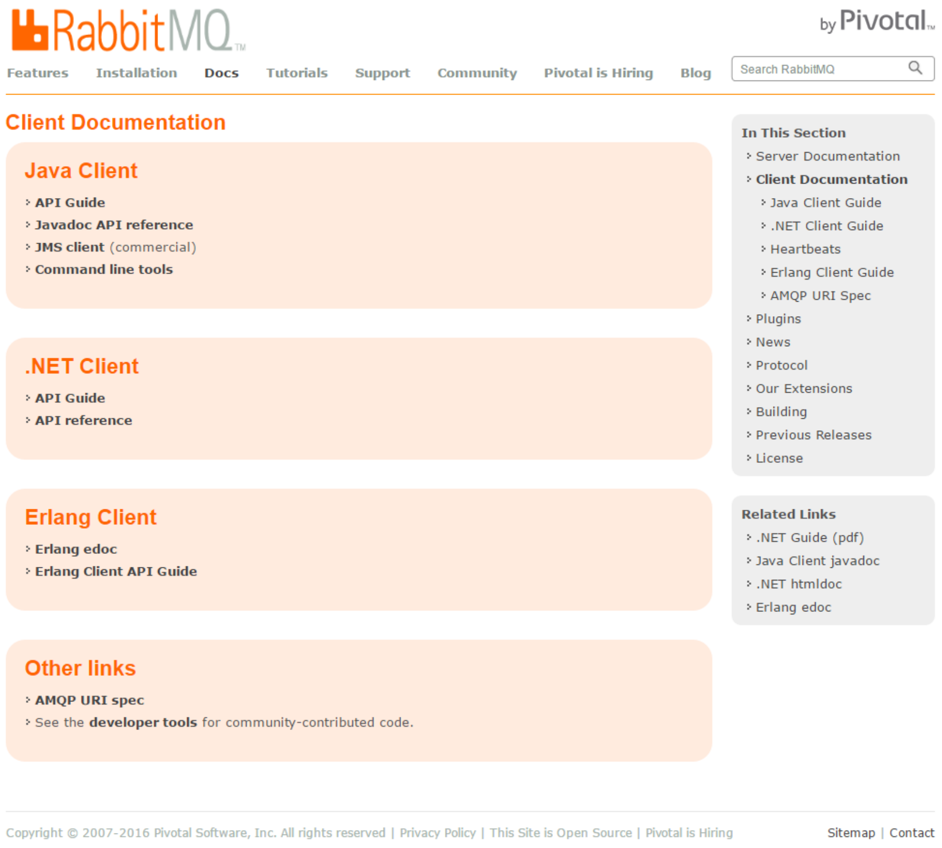


Consumers

Rejecting Messages



RabbitMQ Client Library



RabbitMQ by Pivotal

Features | Installation | Docs | Tutorials | Support | Community | Pivotal is Hiring | Blog | Search RabbitMQ

Client Documentation

Java Client

- API Guide
- Javadoc API reference
- JMS client (commercial)
- Command line tools

.NET Client

- API Guide
- API reference

Erlang Client

- Erlang edoc
- Erlang Client API Guide

Other links

- AMQP URI spec
- See the **developer tools** for community-contributed code.

In This Section

- Server Documentation
- **Client Documentation**
  - Java Client Guide
  - .NET Client Guide
  - Heartbeats
  - Erlang Client Guide
  - AMQP URI Spec
- Plugins
- News
- Protocol
- Our Extensions
- Building
- Previous Releases
- License

Related Links

- .NET Guide (pdf)
- Java Client javadoc
- .NET htmldoc
- Erlang edoc

Copyright © 2007-2016 Pivotal Software, Inc. All rights reserved | Privacy Policy | This Site is Open Source | Pivotal is Hiring — Sitemap | Contact

https://www.rabbitmq.com/clients.html

RabbitMQ Client Library

IModel	*AMQP data channel and provides the AMQP operations*
IConnection	*Represents an AMQP connection*
ConnectionFactory	*Constructs IConnection instances*

ConnectionParameters	*Configures a ConnectionFactory*
QueuingBasicConsumer	*Receives messages delivered from the server*

Connecting to a Message Broker

```
ConnectionFactory factory
   = new ConnectionFactory { HostName = "localhost",
                             UserName = "guest",
                             Password = "guest" };

IConnection connection = factory.CreateConnection();
```

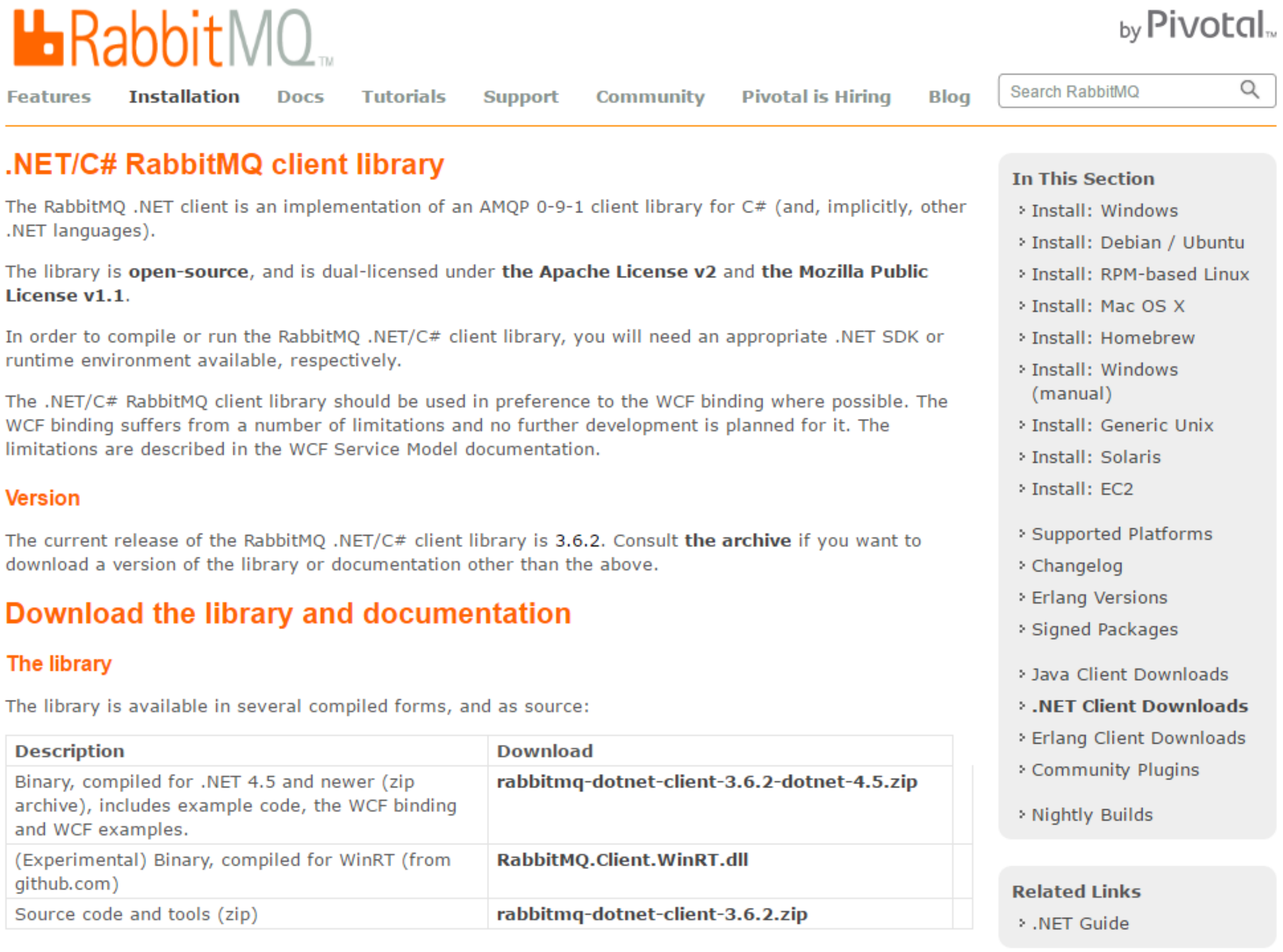
Connecting to a Message Broker

```
IModel _model = connection.CreateModel();
```

Exchanges and Queues

```
channel.ExchangeDeclare(“MyExchange”, "direct");
channel.QueueDeclare(“MyQueue”);

channel.QueueBind(“MyQueue”, ExchangeName, "");
```

Installing the .NET Client Library



.NET/C# RabbitMQ client library

The RabbitMQ .NET client is an implementation of an AMQP 0-9-1 client library for C# (and, implicitly, other .NET languages).

The library is **open-source**, and is dual-licensed under **the Apache License v2** and **the Mozilla Public License v1.1**.

In order to compile or run the RabbitMQ .NET/C# client library, you will need an appropriate .NET SDK or runtime environment available, respectively.

The .NET/C# RabbitMQ client library should be used in preference to the WCF binding where possible. The WCF binding suffers from a number of limitations and no further development is planned for it. The limitations are described in the WCF Service Model documentation.

Version

The current release of the RabbitMQ .NET/C# client library is 3.6.2. Consult **the archive** if you want to download a version of the library or documentation other than the above.

Download the library and documentation

The library

The library is available in several compiled forms, and as source:

Description	Download
Binary, compiled for .NET 4.5 and newer (zip archive), includes example code, the WCF binding and WCF examples.	**rabbitmq-dotnet-client-3.6.2-dotnet-4.5.zip**
(Experimental) Binary, compiled for WinRT (from github.com)	**RabbitMQ.Client.WinRT.dll**
Source code and tools (zip)	**rabbitmq-dotnet-client-3.6.2.zip**

In This Section

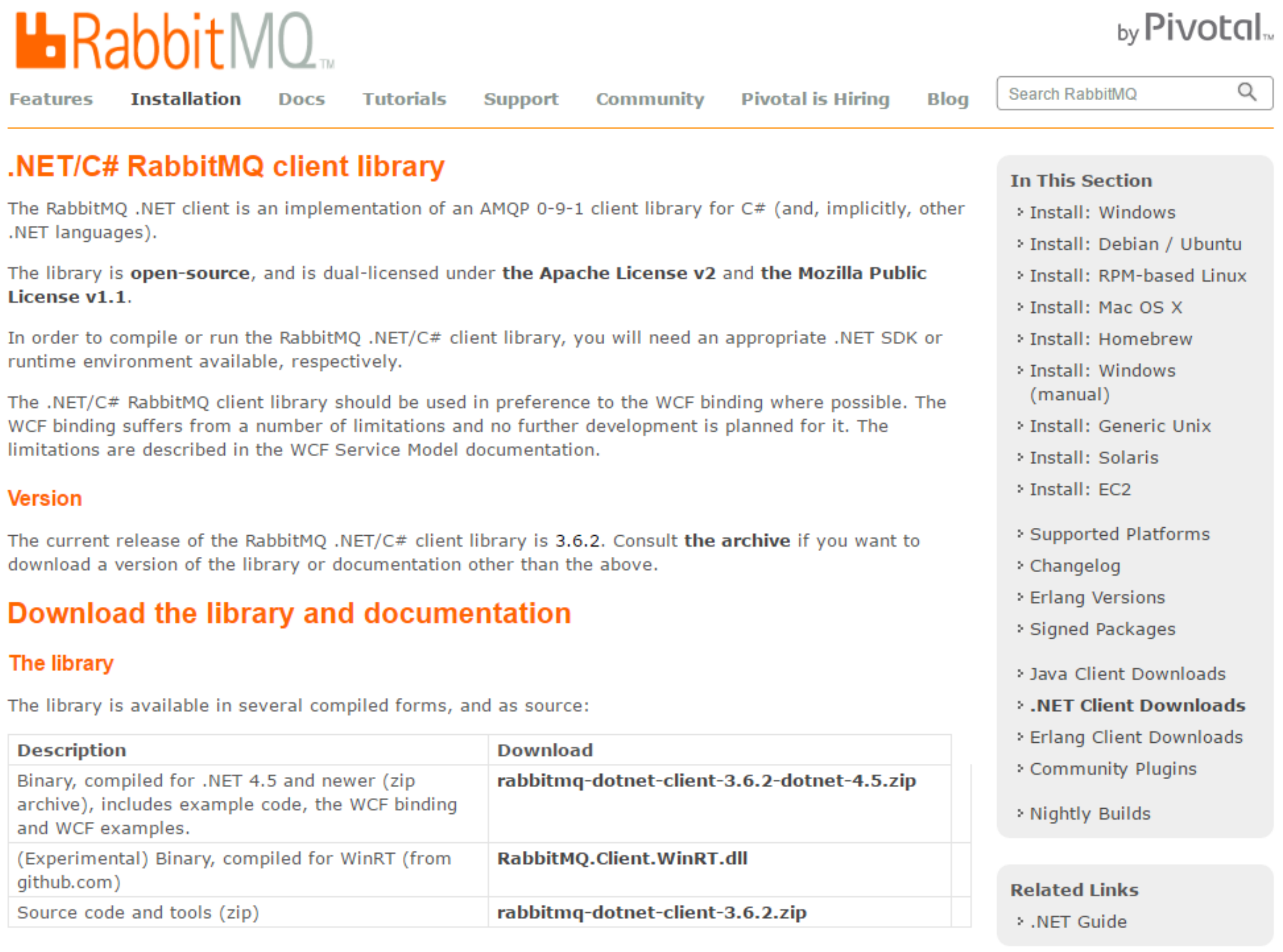
- Install: Windows
- Install: Debian / Ubuntu
- Install: RPM-based Linux
- Install: Mac OS X
- Install: Homebrew
- Install: Windows (manual)
- Install: Generic Unix
- Install: Solaris
- Install: EC2
- Supported Platforms
- Changelog
- Erlang Versions
- Signed Packages
- Java Client Downloads
- **.NET Client Downloads**
- Erlang Client Downloads
- Community Plugins
- Nightly Builds

Related Links

- .NET Guide

https://www.rabbitmq.com/dotnet.html

Installing the .NET Client Library



RabbitMQ by Pivotal

Features | Installation | Docs | Tutorials | Support | Community | Pivotal is Hiring | Blog | Search RabbitMQ

.NET/C# RabbitMQ client library

The RabbitMQ .NET client is an implementation of an AMQP 0-9-1 client library for C# (and, implicitly, other .NET languages).

The library is **open-source**, and is dual-licensed under **the Apache License v2** and **the Mozilla Public License v1.1**.

In order to compile or run the RabbitMQ .NET/C# client library, you will need an appropriate .NET SDK or runtime environment available, respectively.

The .NET/C# RabbitMQ client library should be used in preference to the WCF binding where possible. The WCF binding suffers from a number of limitations and no further development is planned for it. The limitations are described in the WCF Service Model documentation.

Version

The current release of the RabbitMQ .NET/C# client library is 3.6.2. Consult **the archive** if you want to download a version of the library or documentation other than the above.

Download the library and documentation

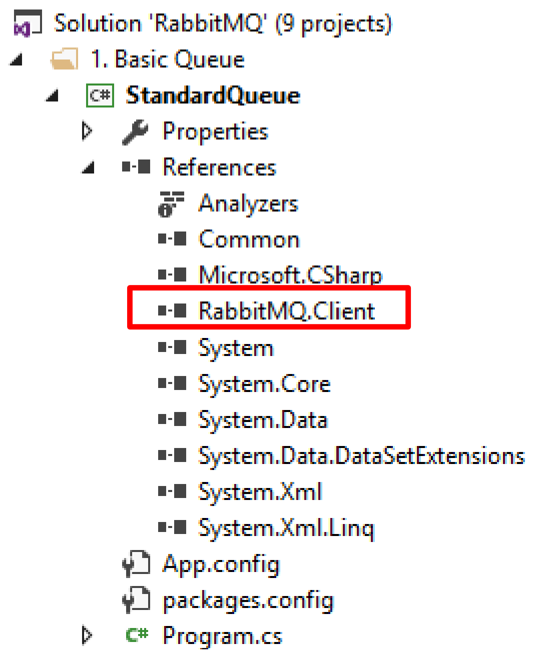
The library

The library is available in several compiled forms, and as source:

Description	Download
Binary, compiled for .NET 4.5 and newer (zip archive), includes example code, the WCF binding and WCF examples.	**rabbitmq-dotnet-client-3.6.2-dotnet-4.5.zip**
(Experimental) Binary, compiled for WinRT (from github.com)	**RabbitMQ.Client.WinRT.dll**
Source code and tools (zip)	**rabbitmq-dotnet-client-3.6.2.zip**

In This Section
- Install: Windows
- Install: Debian / Ubuntu
- Install: RPM-based Linux
- Install: Mac OS X
- Install: Homebrew
- Install: Windows (manual)
- Install: Generic Unix
- Install: Solaris
- Install: EC2
- Supported Platforms
- Changelog
- Erlang Versions
- Signed Packages
- Java Client Downloads
- **.NET Client Downloads**
- Erlang Client Downloads
- Community Plugins
- Nightly Builds

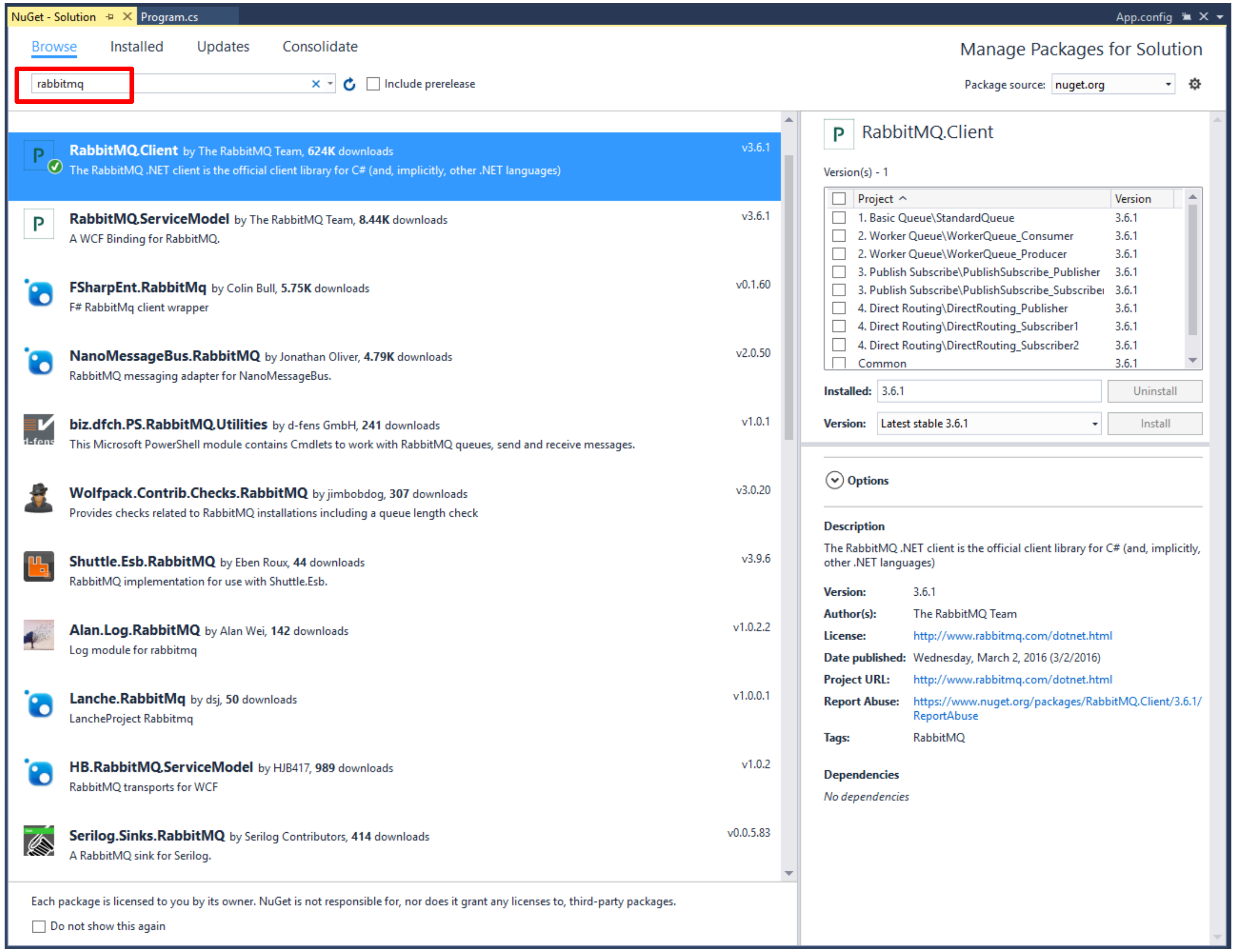
Related Links
- .NET Guide



```
Solution 'RabbitMQ' (9 projects)
◢ 1. Basic Queue
    ◢ StandardQueue
        ▷ Properties
        ◢ References
            Analyzers
            Common
            Microsoft.CSharp
            RabbitMQ.Client
            System
            System.Core
            System.Data
            System.Data.DataSetExtensions
            System.Xml
            System.Xml.Linq
        App.config
        packages.config
        ▷ Program.cs
```

https://www.rabbitmq.com/dotnet.html

Installing the .NET Client Library



rabbitmq

Include prerelease

Package source: nuget.org

RabbitMQ.Client by The RabbitMQ Team, **624K** downloads — v3.6.1
The RabbitMQ .NET client is the official client library for C# (and, implicitly, other .NET languages)

RabbitMQ.ServiceModel by The RabbitMQ Team, **8.44K** downloads — v3.6.1
A WCF Binding for RabbitMQ.

FSharpEnt.RabbitMq by Colin Bull, **5.75K** downloads — v0.1.60
F# RabbitMq client wrapper

NanoMessageBus.RabbitMQ by Jonathan Oliver, **4.79K** downloads — v2.0.50
RabbitMQ messaging adapter for NanoMessageBus.

biz.dfch.PS.RabbitMQ.Utilities by d-fens GmbH, **241** downloads — v1.0.1
This Microsoft PowerShell module contains Cmdlets to work with RabbitMQ queues, send and receive messages.

Wolfpack.Contrib.Checks.RabbitMQ by jimbobdog, **307** downloads — v3.0.20
Provides checks related to RabbitMQ installations including a queue length check

Shuttle.Esb.RabbitMQ by Eben Roux, **44** downloads — v3.9.6
RabbitMQ implementation for use with Shuttle.Esb.

Alan.Log.RabbitMQ by Alan Wei, **142** downloads — v1.0.2.2
Log module for rabbitmq

Lanche.RabbitMq by dsj, **50** downloads — v1.0.0.1
LancheProject Rabbitmq

HB.RabbitMQ.ServiceModel by HJB417, **989** downloads — v1.0.2
RabbitMQ transports for WCF

Serilog.Sinks.RabbitMQ by Serilog Contributors, **414** downloads — v0.0.5.83
A RabbitMQ sink for Serilog.

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

Do not show this again

RabbitMQ.Client

Version(s) - 1

Project	Version
1. Basic Queue\StandardQueue	3.6.1
2. Worker Queue\WorkerQueue_Consumer	3.6.1
2. Worker Queue\WorkerQueue_Producer	3.6.1
3. Publish Subscribe\PublishSubscribe_Publisher	3.6.1
3. Publish Subscribe\PublishSubscribe_Subscriber	3.6.1
4. Direct Routing\DirectRouting_Publisher	3.6.1
4. Direct Routing\DirectRouting_Subscriber1	3.6.1
4. Direct Routing\DirectRouting_Subscriber2	3.6.1
Common	3.6.1

Installed: 3.6.1

Version: Latest stable 3.6.1

Description

The RabbitMQ .NET client is the official client library for C# (and, implicitly, other .NET languages)

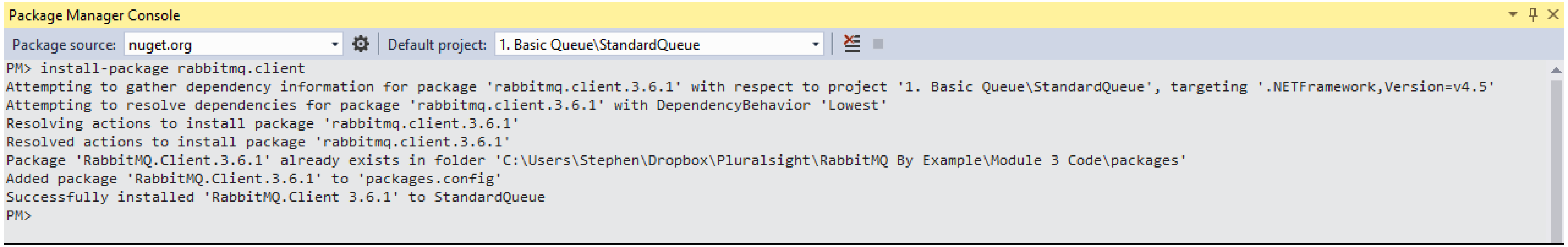
Version:	3.6.1
Author(s):	The RabbitMQ Team
License:	http://www.rabbitmq.com/dotnet.html
Date published:	Wednesday, March 2, 2016 (3/2/2016)
Project URL:	http://www.rabbitmq.com/dotnet.html
Report Abuse:	https://www.nuget.org/packages/RabbitMQ.Client/3.6.1/ReportAbuse
Tags:	RabbitMQ

Dependencies

No dependencies

Installing the .NET Client Library

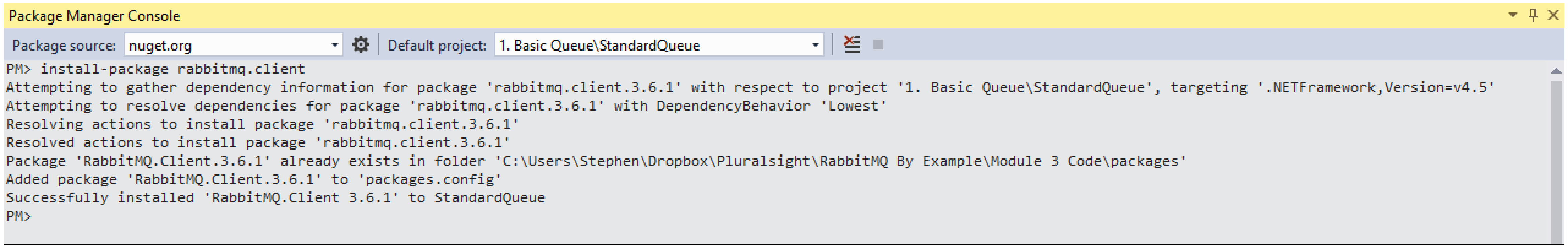
```
Install-Package RabbitMQ.Client
```


Package Manager Console
Package source: nuget.org
Default project: 1. Basic Queue\StandardQueue

```
PM> install-package rabbitmq.client
Attempting to gather dependency information for package 'rabbitmq.client.3.6.1' with respect to project '1. Basic Queue\StandardQueue', targeting '.NETFramework,Version=v4.5'
Attempting to resolve dependencies for package 'rabbitmq.client.3.6.1' with DependencyBehavior 'Lowest'
Resolving actions to install package 'rabbitmq.client.3.6.1'
Resolved actions to install package 'rabbitmq.client.3.6.1'
Package 'RabbitMQ.Client.3.6.1' already exists in folder 'C:\Users\Stephen\Dropbox\Pluralsight\RabbitMQ By Example\Module 3 Code\packages'
Added package 'RabbitMQ.Client.3.6.1' to 'packages.config'
Successfully installed 'RabbitMQ.Client 3.6.1' to StandardQueue
PM>
```

Installing the .NET Client Library

```
Install-Package RabbitMQ.Client
```

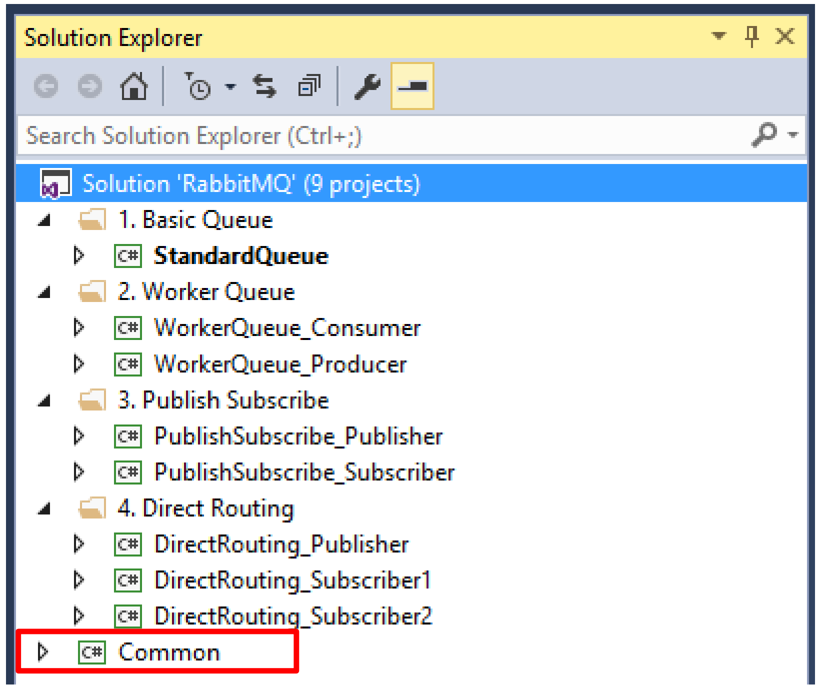


Package Manager Console

Package source: nuget.org   Default project: 1. Basic Queue\StandardQueue

```
PM> install-package rabbitmq.client
Attempting to gather dependency information for package 'rabbitmq.client.3.6.1' with respect to project '1. Basic Queue\StandardQueue', targeting '.NETFramework,Version=v4.5'
Attempting to resolve dependencies for package 'rabbitmq.client.3.6.1' with DependencyBehavior 'Lowest'
Resolving actions to install package 'rabbitmq.client.3.6.1'
Resolved actions to install package 'rabbitmq.client.3.6.1'
Package 'RabbitMQ.Client.3.6.1' already exists in folder 'C:\Users\Stephen\Dropbox\Pluralsight\RabbitMQ By Example\Module 3 Code\packages'
Added package 'RabbitMQ.Client.3.6.1' to 'packages.config'
Successfully installed 'RabbitMQ.Client 3.6.1' to StandardQueue
PM>
```

Common Code



Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'RabbitMQ' (9 projects)
- 1. Basic Queue
  - StandardQueue
- 2. Worker Queue
  - WorkerQueue_Consumer
  - WorkerQueue_Producer
- 3. Publish Subscribe
  - PublishSubscribe_Publisher
  - PublishSubscribe_Subscriber
- 4. Direct Routing
  - DirectRouting_Publisher
  - DirectRouting_Subscriber1
  - DirectRouting_Subscriber2
- Common

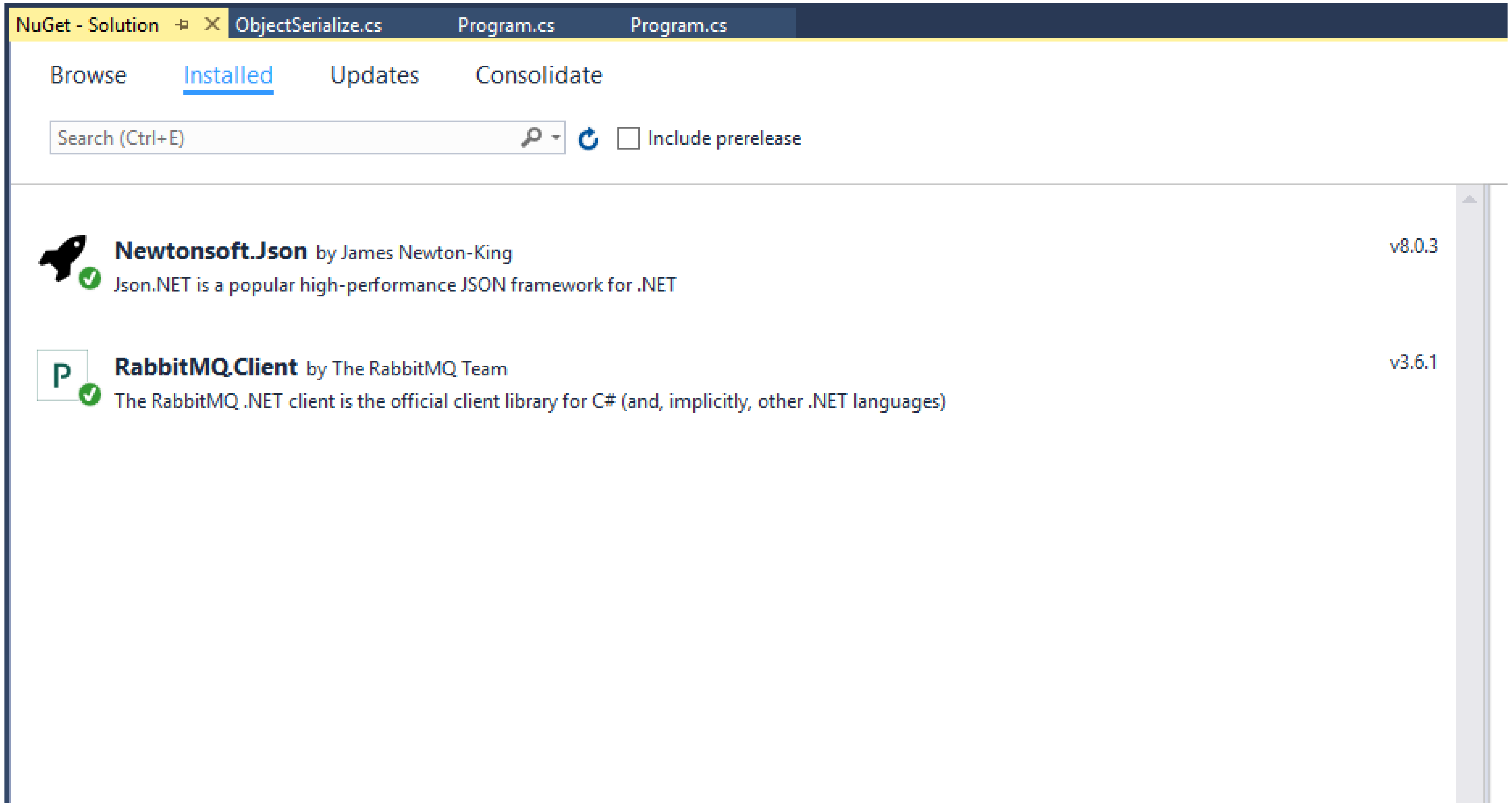
Payment.cs

```
using System;

namespace RabbitMQ.Examples

{

    [Serializable]

    public class Payment

    {

        public decimal AmountToPay;

        public string CardNumber;

        public string Name;

    }

}
```

PurchaseOrder.cs

```
using System;

namespace RabbitMQ.Examples

{

    [Serializable]

    public class PurchaseOrder

    {

        public decimal AmountToPay;

        public string PoNumber;

        public string CompanyName;

        public int PaymentDayTerms;

    }

}
```

ObjectSerialize.cs


NuGet - Solution ObjectSerialize.cs Program.cs Program.cs

Browse Installed Updates Consolidate

Search (Ctrl+E) Include prerelease

Newtonsoft.Json by James Newton-King v8.0.3
Json.NET is a popular high-performance JSON framework for .NET

RabbitMQ.Client by The RabbitMQ Team v3.6.1
The RabbitMQ .NET client is the official client library for C# (and, implicitly, other .NET languages)

ObjectSerialize.cs

```
public static byte[] Serialize(this Object obj)
{
    if (obj == null)
    {
        return null;
    }

    var json = JsonConvert.SerializeObject(obj);
    return Encoding.ASCII.GetBytes(json);
}
```

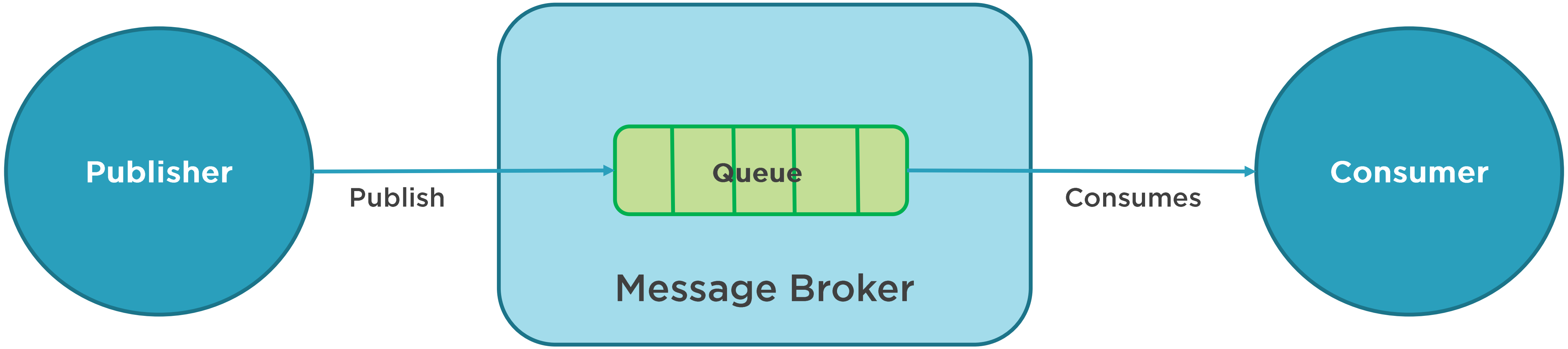
ObjectSerialize.cs

```
public static Object DeSerialize(this byte[] arrBytes, Type type)
{
    var json = Encoding.Default.GetString(arrBytes);

    return JsonConvert.DeserializeObject(json, type);
}
```

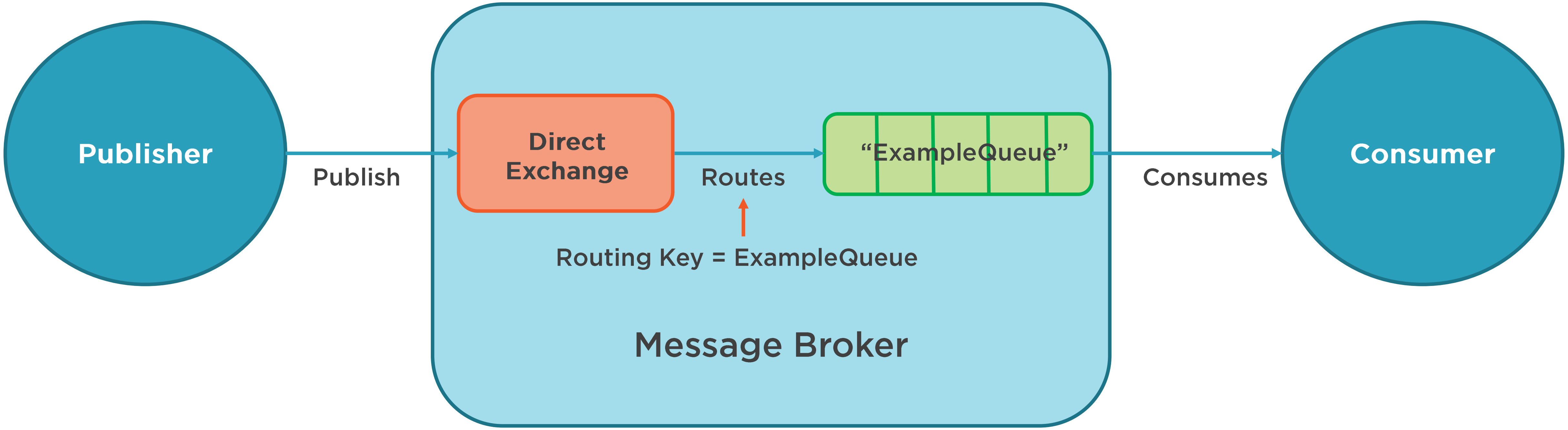
ObjectSerialize.cs

```
Payment payment1 = new Payment {
                                  AmountToPay = 25.0m,
                                  CardNumber = "1234123412341234"
                               };

byte [] serialized = payment1.Serialize();

Payment payment_deserialized =  serialized.DeSerialize();
```

Standard Queue



Standard Queue



Standard Queue

```
var payment1 = new Payment { AmountToPay = 25.0m,
                             CardNumber = "1234123412341234" };
var payment2 = new Payment { AmountToPay = 5.0m,
                              CardNumber = "1234123412341234" };


CreateQueue();
SendMessage(payment1);
SendMessage(payment2);
Recieve();
```

Standard Queue

```
private static void SendMessage(Payment message)
{
    _model.BasicPublish("", QueueName, null, message.Serialize());

    Console.WriteLine(" [x] Payment Message Sent : {0} : {1}",
                        message.CardNumber,
                        message.AmountToPay);
}
```

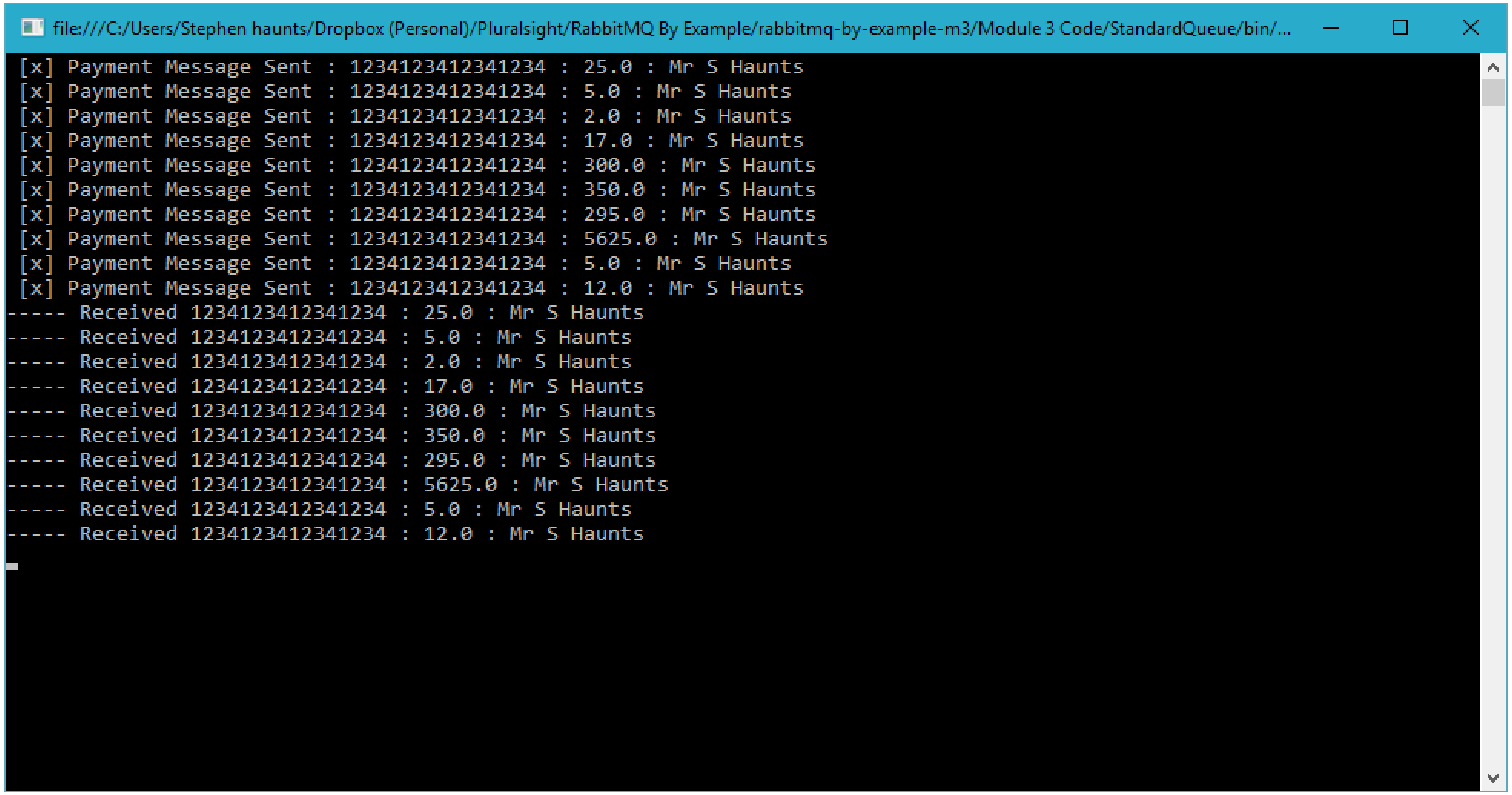
Standard Queue

```
public static void Recieve()
{
    var consumer = new QueueingBasicConsumer(_model);
    var msgCount = GetMessageCount(_model, QueueName);
    _model.BasicConsume(QueueName, true, consumer);

    var count = 0;
    while (count < msgCount){
        var message = (Payment)consumer.Queue.Dequeue().Body.DeSerialize();
        count++;
    }  }
```

Standard Queue

```
private static uint GetMessageCount(IModel channel, string queueName)
{
    var results = channel.QueueDeclare(queueName, true, false, false, null);
    return results.MessageCount;
}
```

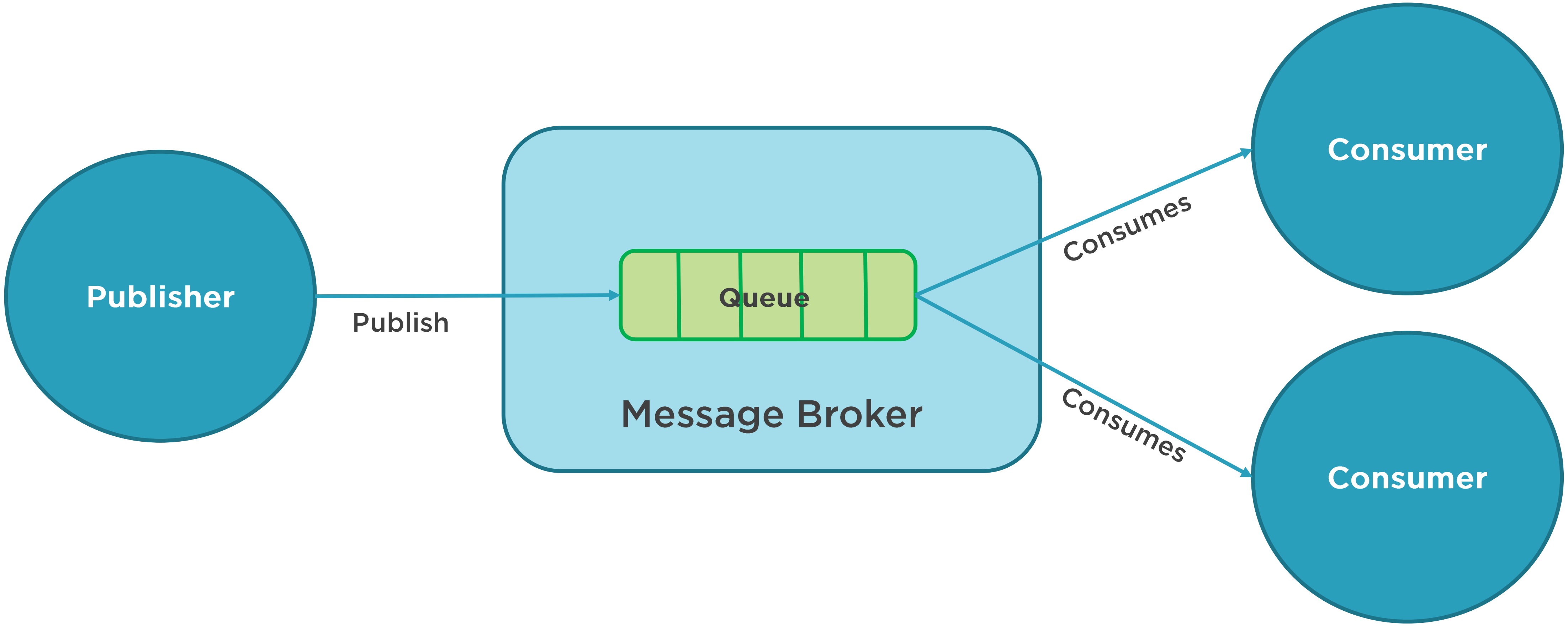
Standard Queue



file:///C:/Users/Stephen haunts/Dropbox (Personal)/Pluralsight/RabbitMQ By Example/rabbitmq-by-example-m3/Module 3 Code/StandardQueue/bin/...

```
 [x] Payment Message Sent : 1234123412341234 : 25.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 5.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 2.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 17.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 300.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 350.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 295.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 5625.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 5.0 : Mr S Haunts
 [x] Payment Message Sent : 1234123412341234 : 12.0 : Mr S Haunts
----- Received 1234123412341234 : 25.0 : Mr S Haunts
----- Received 1234123412341234 : 5.0 : Mr S Haunts
----- Received 1234123412341234 : 2.0 : Mr S Haunts
----- Received 1234123412341234 : 17.0 : Mr S Haunts
----- Received 1234123412341234 : 300.0 : Mr S Haunts
----- Received 1234123412341234 : 350.0 : Mr S Haunts
----- Received 1234123412341234 : 295.0 : Mr S Haunts
----- Received 1234123412341234 : 5625.0 : Mr S Haunts
----- Received 1234123412341234 : 5.0 : Mr S Haunts
----- Received 1234123412341234 : 12.0 : Mr S Haunts
```

Worker Queue



Worker Queue

```
var payment1 = new Payment { AmountToPay = 25.0m,
                             CardNumber = "1234123412341234" };
var payment2 = new Payment { AmountToPay = 5.0m,
                              CardNumber = "1234123412341234" };


CreateConnection();


SendMessage(payment1);
SendMessage(payment2);
```

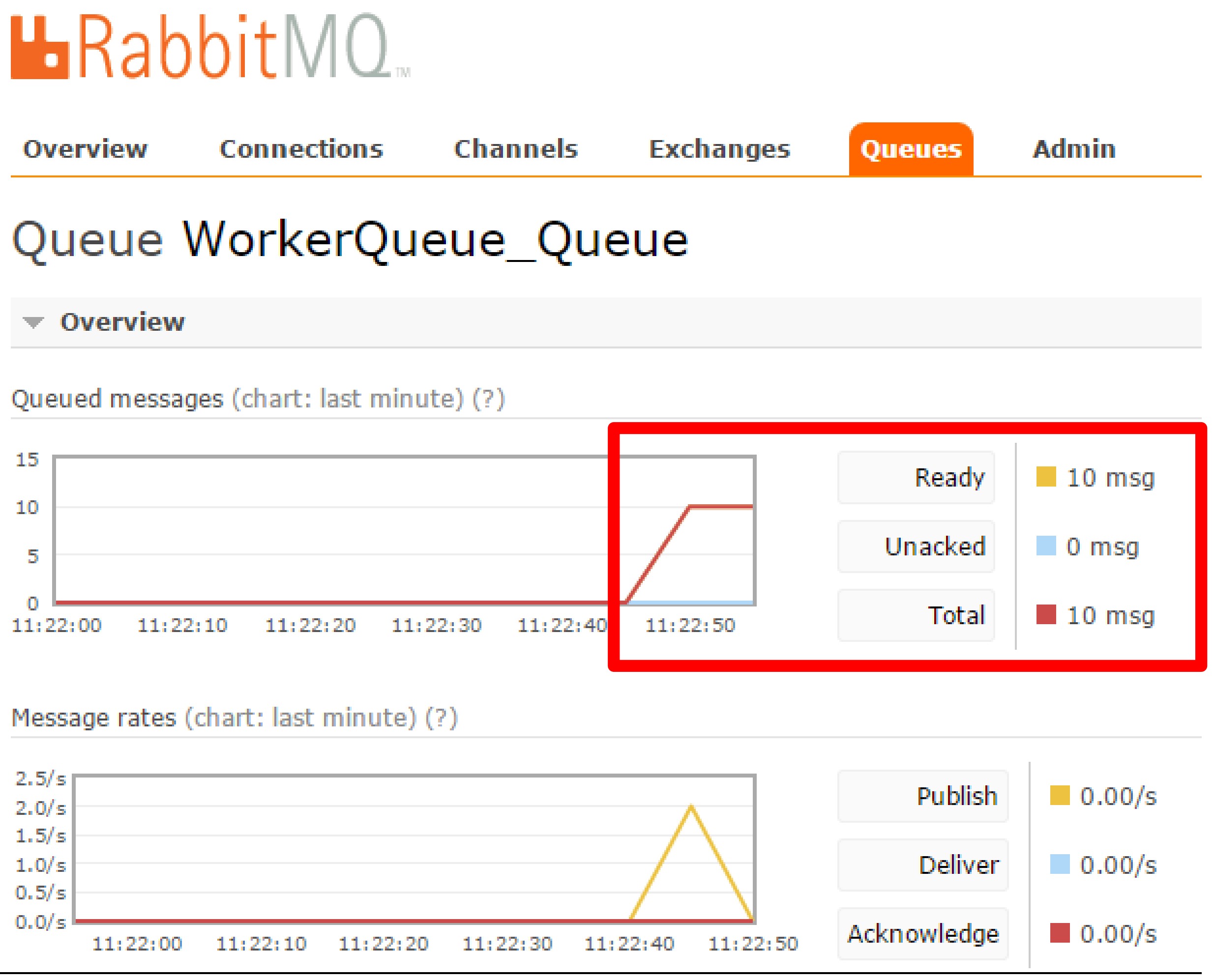
Worker Queue

```
private const string QueueName = "WorkerQueue_Queue";
private static void CreateConnection()
{
    _factory = new ConnectionFactory { HostName = "localhost",
                                       UserName = "guest",
                                       Password = "guest" };
    _connection = _factory.CreateConnection();
    _model = _connection.CreateModel();
    _model.QueueDeclare(QueueName, true, false, false, null);
}
```

Worker Queue

```
private static void SendMessage(Payment message)
{
    _model.BasicPublish("", QueueName, null, message.Serialize());
}
```

Worker Queue



Queue WorkerQueue_Queue

Overview

Queued messages (chart: last minute) (?)

Ready	🟨 10 msg
Unacked	🟦 0 msg
Total	🟥 10 msg

Message rates (chart: last minute) (?)

Publish	🟨 0.00/s
Deliver	🟦 0.00/s
Acknowledge	🟥 0.00/s

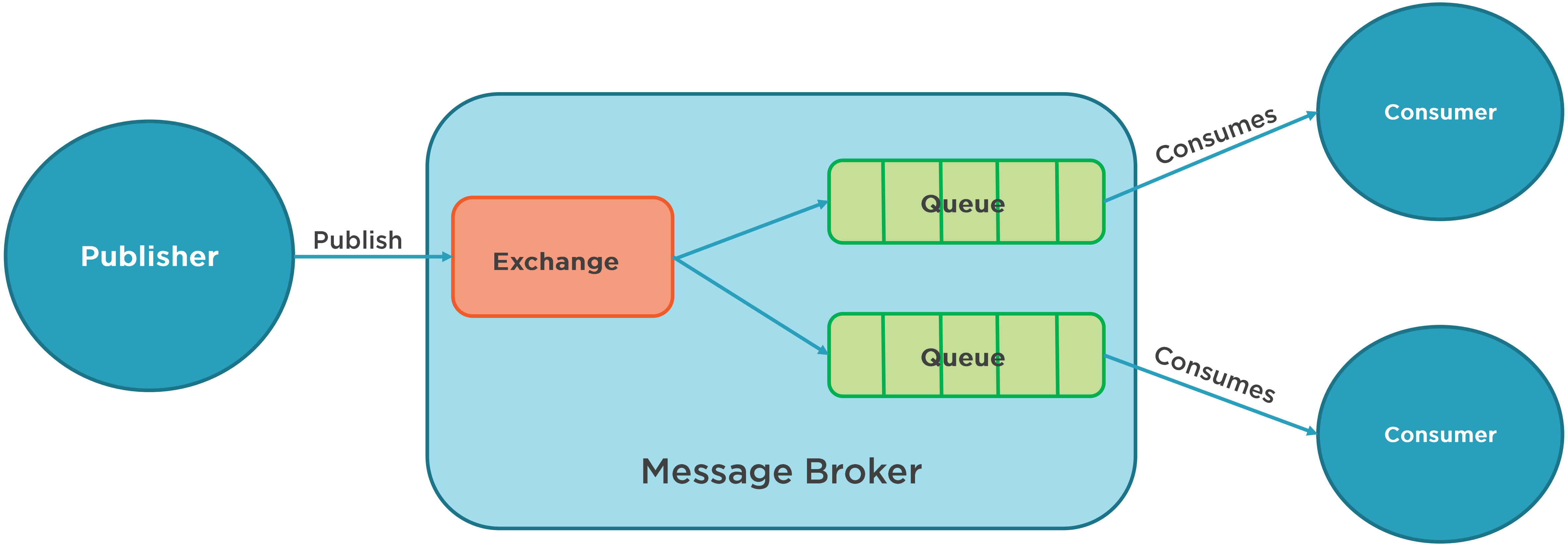
Worker Queue

```
public static void Receive()
{
    _factory = new ConnectionFactory { HostName = "localhost",
                                       UserName = "guest",
                                       Password = "guest" };
    using (_connection = _factory.CreateConnection())
    {
        using (var channel = _connection.CreateModel())
        {
            channel.QueueDeclare(QueueName, true, false, false, null);
            channel.BasicQos(0, 1, false);
```

Worker Queue

```
var consumer = new QueueingBasicConsumer(channel);
channel.BasicConsume(QueueName, false, consumer);
while (true)
{
    var ea = consumer.Queue.Dequeue();
    var message = (Payment)ea.Body.DeSerialize();
    channel.BasicAck(ea.DeliveryTag, false);

    Console.WriteLine(" Processed {0} : {1}", message.CardNumber,
                                                message.AmountToPay);
}
```

Publish and Subscribe



Publish and Subscribe

```
var payment1 = new Payment { AmountToPay = 25.0m,
                             CardNumber = "1234123412341234" };

CreateConnection();

SendPayment(payment1);
```

Publish and Subscribe

```
_factory = new ConnectionFactory { HostName = "localhost", UserName = "guest",
                                   Password = "guest" };
_connection = _factory.CreateConnection();
_model = _connection.CreateModel();
_model.ExchangeDeclare(ExchangeName, “fanout“, true);
```

Publish and Subscribe

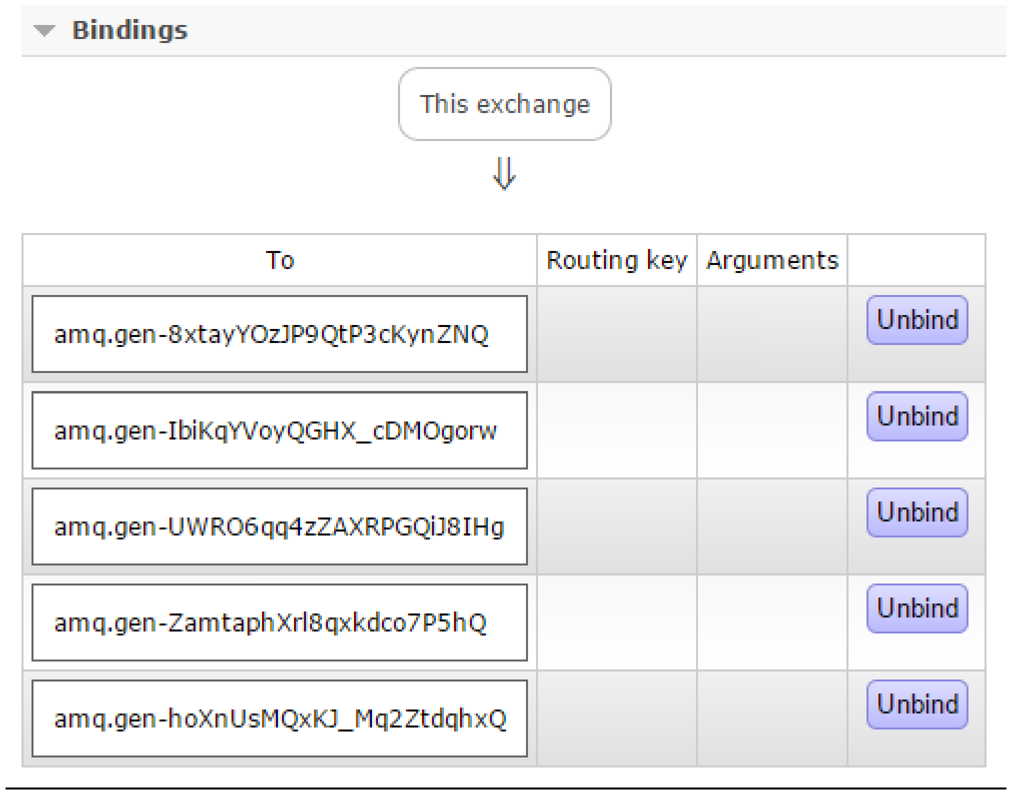
```
private static void SendMessage(Payment message)
{
    _model.BasicPublish(ExchangeName, "", null, message.Serialize());

    Console.WriteLine(" Payment Sent {0}, £{1}", message.CardNumber,
                                                 message.AmountToPay);
}
```

Publish and Subscribe

```
private static string DeclareAndBindQueueToExchange(IModel channel)
{
    channel.ExchangeDeclare(ExchangeName, "fanout");

    var queueName = channel.QueueDeclare().QueueName;
    channel.QueueBind(queueName, ExchangeName, "");
    _consumer = new QueueingBasicConsumer(channel);

    return queueName;
}
```

amq.gen-TsdoX9qziswm9QCbdkp9Zw

Publish and Subscribe



Bindings

This exchange

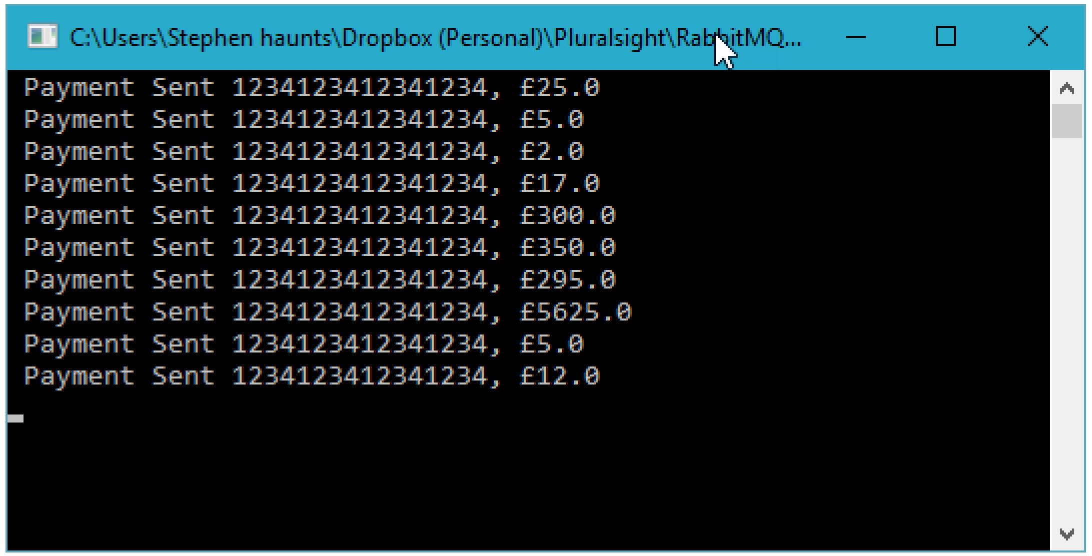
⇓

To	Routing key	Arguments	
amq.gen-8xtayYOzJP9QtP3cKynZNQ			Unbind
amq.gen-IbiKqYVoyQGHX_cDMOgorw			Unbind
amq.gen-UWRO6qq4zZAXRPGQiJ8IHg			Unbind
amq.gen-ZamtaphXrl8qxkdco7P5hQ			Unbind
amq.gen-hoXnUsMQxKJ_Mq2ZtdqhxQ			Unbind

Publish and Subscribe

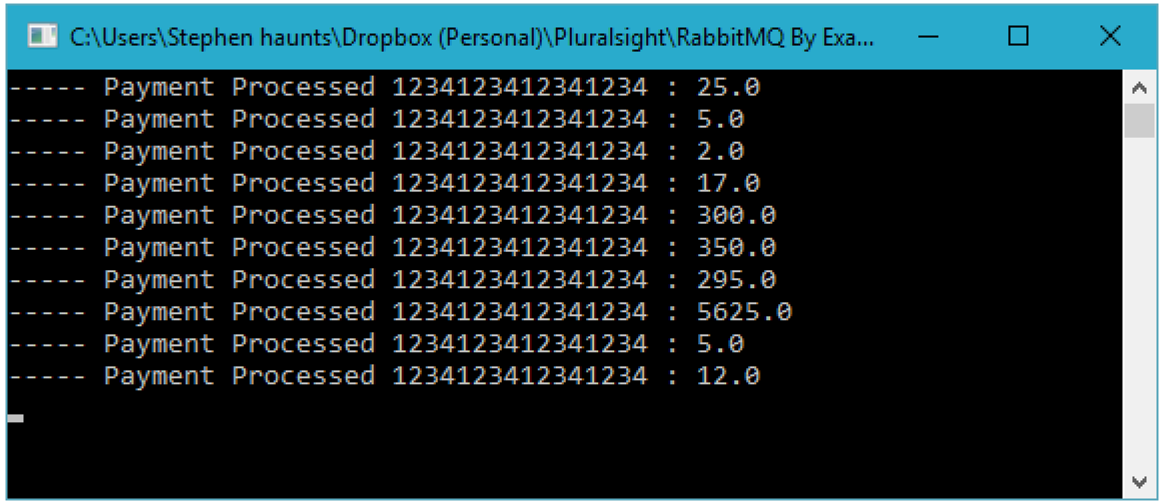
```
channel.BasicConsume(queueName, true, _consumer);
while (true)
{
    var ea = _consumer.Queue.Dequeue();
    var message = (Payment)ea.Body.DeSerialize(typeof(Payment));

    Console.WriteLine("----- Payment Processed {0} : {1}",
                                 message.CardNumber,
                                 message.AmountToPay);
}
```

Publish and Subscribe



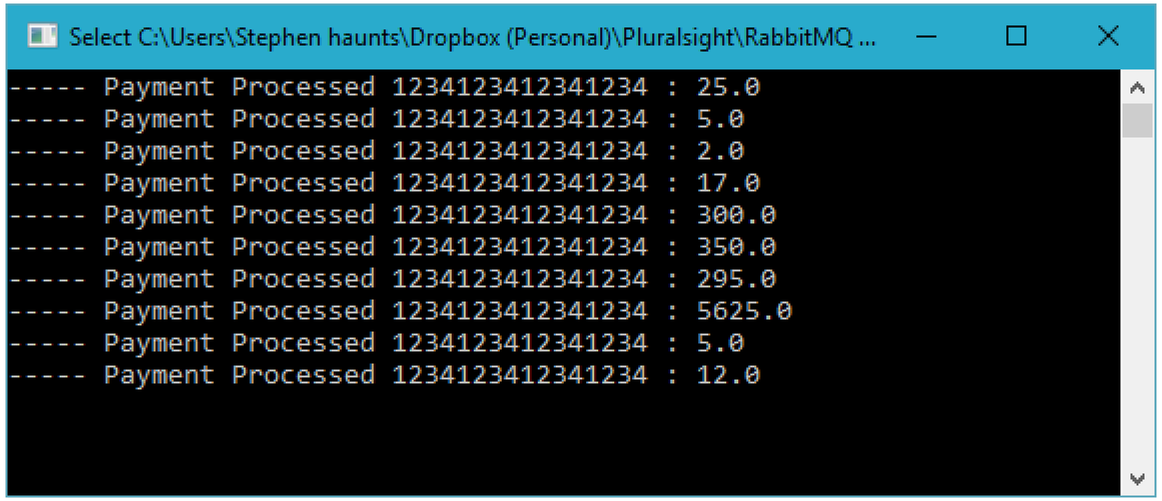
C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ...

```
Payment Sent 1234123412341234, £25.0
Payment Sent 1234123412341234, £5.0
Payment Sent 1234123412341234, £2.0
Payment Sent 1234123412341234, £17.0
Payment Sent 1234123412341234, £300.0
Payment Sent 1234123412341234, £350.0
Payment Sent 1234123412341234, £295.0
Payment Sent 1234123412341234, £5625.0
Payment Sent 1234123412341234, £5.0
Payment Sent 1234123412341234, £12.0
```



C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ By Exa...

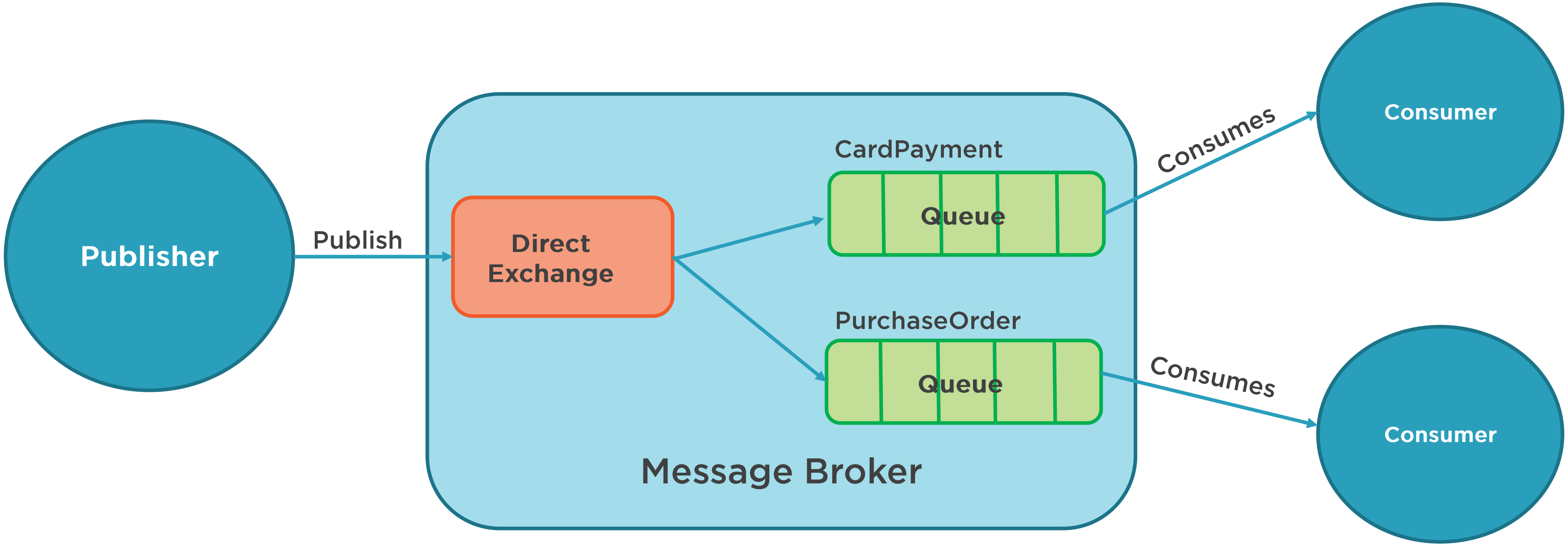
```
----- Payment Processed 1234123412341234 : 25.0
----- Payment Processed 1234123412341234 : 5.0
----- Payment Processed 1234123412341234 : 2.0
----- Payment Processed 1234123412341234 : 17.0
----- Payment Processed 1234123412341234 : 300.0
----- Payment Processed 1234123412341234 : 350.0
----- Payment Processed 1234123412341234 : 295.0
----- Payment Processed 1234123412341234 : 5625.0
----- Payment Processed 1234123412341234 : 5.0
----- Payment Processed 1234123412341234 : 12.0
```



Select C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ ...

```
----- Payment Processed 1234123412341234 : 25.0
----- Payment Processed 1234123412341234 : 5.0
----- Payment Processed 1234123412341234 : 2.0
----- Payment Processed 1234123412341234 : 17.0
----- Payment Processed 1234123412341234 : 300.0
----- Payment Processed 1234123412341234 : 350.0
----- Payment Processed 1234123412341234 : 295.0
----- Payment Processed 1234123412341234 : 5625.0
----- Payment Processed 1234123412341234 : 5.0
----- Payment Processed 1234123412341234 : 12.0
```

Direct Routing



Direct Routing

```
var payment1 = new Payment { AmountToPay = 25.0m,
                             CardNumber = "1234123412341234" };


var purchaseOrder1 = new PurchaseOrder{AmountToPay = 50.0m,
                                      CompanyName = "Company A",
                                      PaymentDayTerms = 75,
                                      PoNumber = "123434A"};
CreateConnection();
SendPayment(payment1);
SendPurchaseOrder(purchaseOrder1);
```

Direct Routing

```
 _factory = new ConnectionFactory { HostName = "localhost",
                                    UserName = "guest", Password = "guest" };
_connection = _factory.CreateConnection();
_model = _connection.CreateModel();

_model.ExchangeDeclare(ExchangeName, "direct");
_model.QueueDeclare(CardPaymentQueueName, true, false, false, null);
_model.QueueDeclare(PurchaseOrderQueueName, true, false, false, null);

_model.QueueBind(CardPaymentQueueName, ExchangeName, "CardPayment");
_model.QueueBind(PurchaseOrderQueueName, ExchangeName, "PurchaseOrder");
```

Direct Routing

```
private const string ExchangeName = "DirectRouting_Exchange";

private const string CardPaymentQueueName = "CardPaymentDirectRouting_Queue";


_factory = new ConnectionFactory { HostName = "localhost",

                                   UserName = "guest",

                                   Password = "guest" };

    using (_connection = _factory.CreateConnection())

    {

        using (var channel = _connection.CreateModel())

        {
```

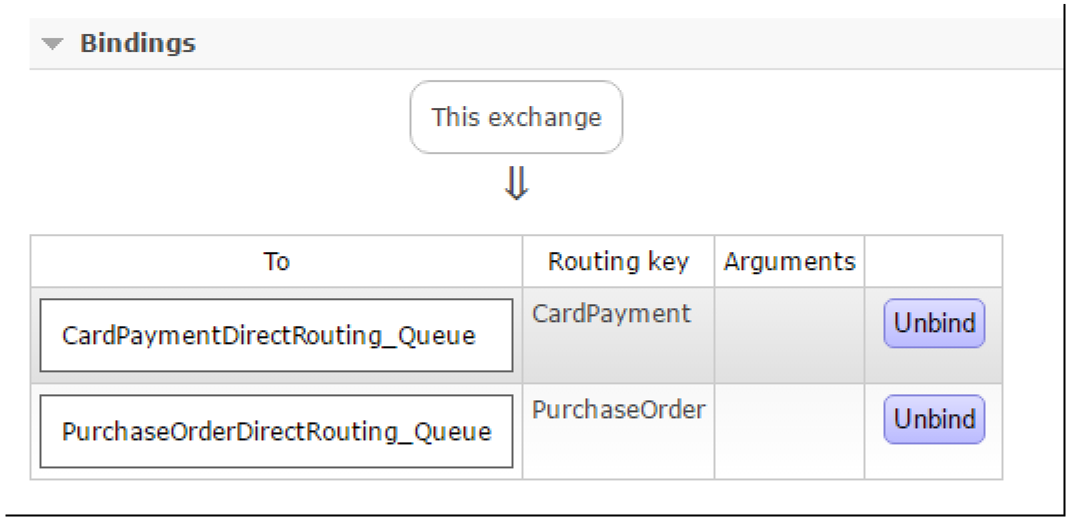
Direct Routing

```
channel.ExchangeDeclare(ExchangeName, "direct");
channel.QueueDeclare(CardPaymentQueueName, true, false, false, null);
channel.QueueBind(CardPaymentQueueName, ExchangeName, "CardPayment");

channel.BasicQos(0, 1, false);

var consumer = new QueueingBasicConsumer(channel);
channel.BasicConsume(CardPaymentQueueName, false, consumer);
```

Direct Routing

```
while (true)
{
    var ea = consumer.Queue.Dequeue();
    var message = (Payment)ea.Body.DeSerialize(typeof(Payment));
    var routingKey = ea.RoutingKey;
    channel.BasicAck(ea.DeliveryTag, false);

    Console.WriteLine("--- Payment - Routing Key <{0}> : {1} : {2}",
                      routingKey, message.CardNumber, message.AmountToPay);
}
```

Direct Routing



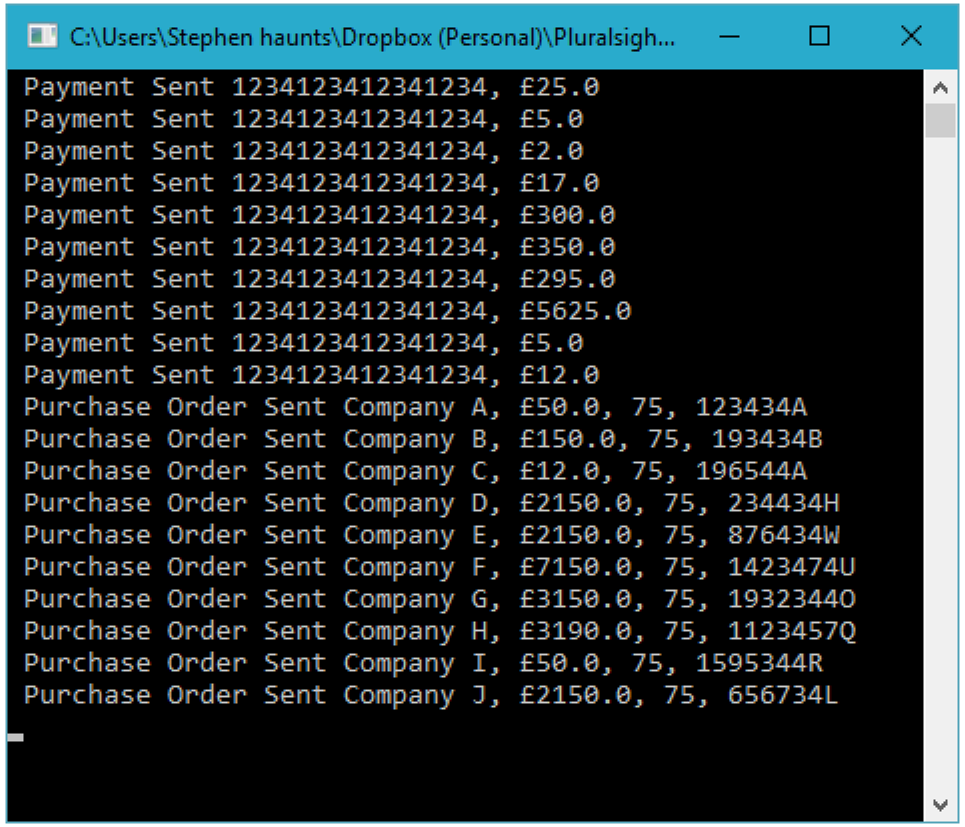
Bindings

This exchange

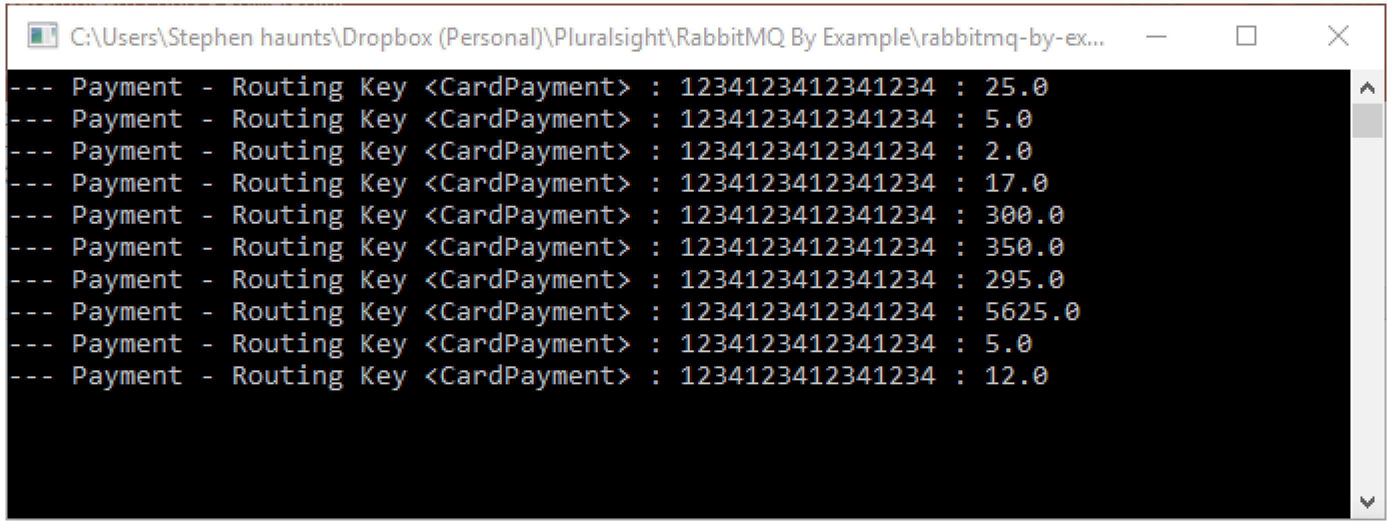
⇓

To	Routing key	Arguments	
CardPaymentDirectRouting_Queue	CardPayment		Unbind
PurchaseOrderDirectRouting_Queue	PurchaseOrder		Unbind

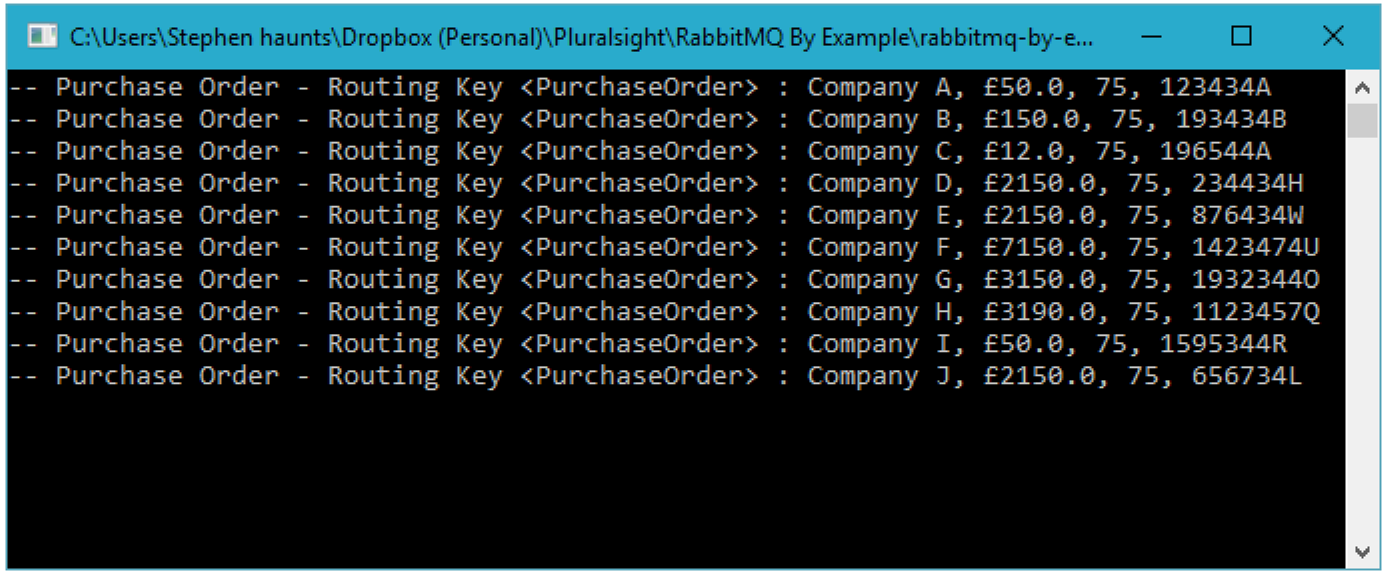
Direct Routing



```
Payment Sent 1234123412341234, £25.0
Payment Sent 1234123412341234, £5.0
Payment Sent 1234123412341234, £2.0
Payment Sent 1234123412341234, £17.0
Payment Sent 1234123412341234, £300.0
Payment Sent 1234123412341234, £350.0
Payment Sent 1234123412341234, £295.0
Payment Sent 1234123412341234, £5625.0
Payment Sent 1234123412341234, £5.0
Payment Sent 1234123412341234, £12.0
Purchase Order Sent Company A, £50.0, 75, 123434A
Purchase Order Sent Company B, £150.0, 75, 193434B
Purchase Order Sent Company C, £12.0, 75, 196544A
Purchase Order Sent Company D, £2150.0, 75, 234434H
Purchase Order Sent Company E, £2150.0, 75, 876434W
Purchase Order Sent Company F, £7150.0, 75, 1423474U
Purchase Order Sent Company G, £3150.0, 75, 1932344O
Purchase Order Sent Company H, £3190.0, 75, 1123457Q
Purchase Order Sent Company I, £50.0, 75, 1595344R
Purchase Order Sent Company J, £2150.0, 75, 656734L
```

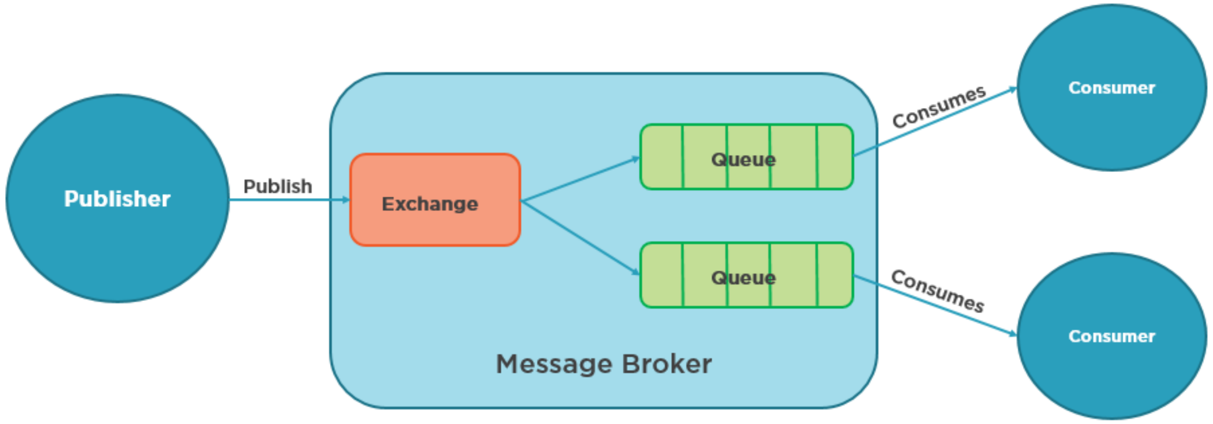

C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ By Example\rabbitmq-by-ex...

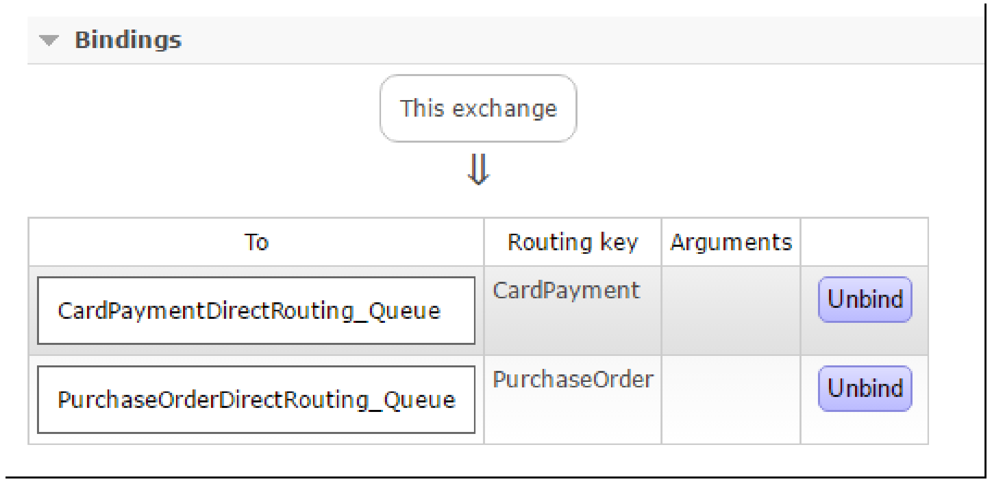
```
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 25.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 5.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 2.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 17.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 300.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 350.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 295.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 5625.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 5.0
--- Payment - Routing Key <CardPayment> : 1234123412341234 : 12.0
```


C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ By Example\rabbitmq-by-e...

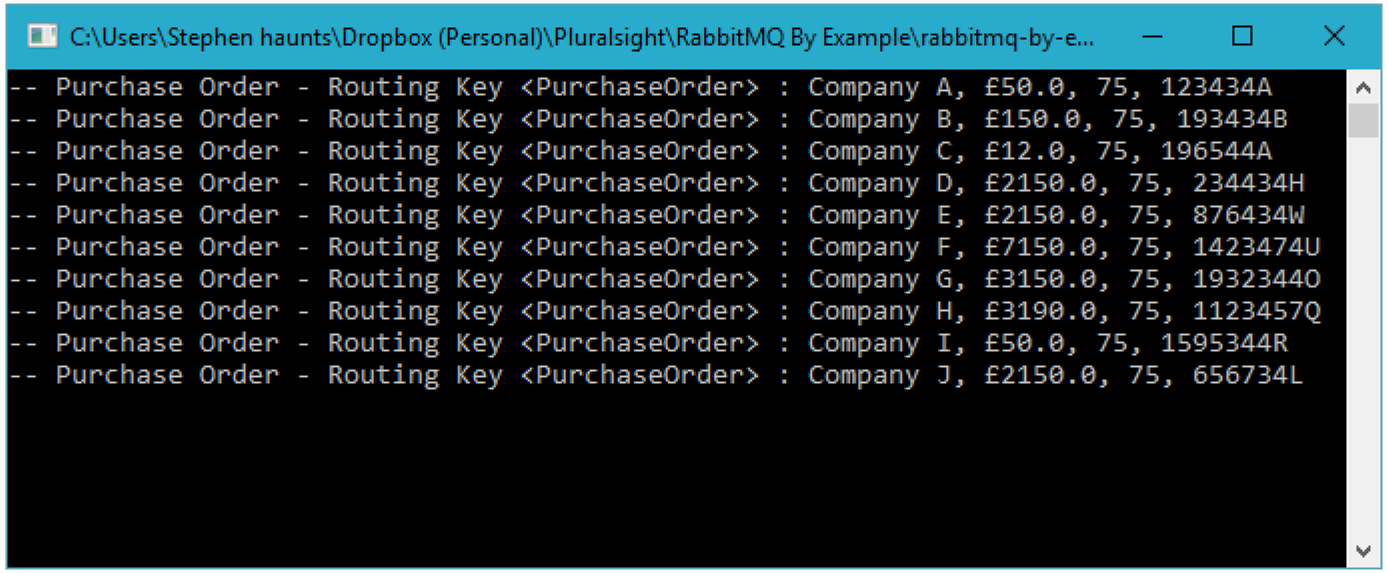
```
-- Purchase Order - Routing Key <PurchaseOrder> : Company A, £50.0, 75, 123434A
-- Purchase Order - Routing Key <PurchaseOrder> : Company B, £150.0, 75, 193434B
-- Purchase Order - Routing Key <PurchaseOrder> : Company C, £12.0, 75, 196544A
-- Purchase Order - Routing Key <PurchaseOrder> : Company D, £2150.0, 75, 234434H
-- Purchase Order - Routing Key <PurchaseOrder> : Company E, £2150.0, 75, 876434W
-- Purchase Order - Routing Key <PurchaseOrder> : Company F, £7150.0, 75, 1423474U
-- Purchase Order - Routing Key <PurchaseOrder> : Company G, £3150.0, 75, 1932344O
-- Purchase Order - Routing Key <PurchaseOrder> : Company H, £3190.0, 75, 1123457Q
-- Purchase Order - Routing Key <PurchaseOrder> : Company I, £50.0, 75, 1595344R
-- Purchase Order - Routing Key <PurchaseOrder> : Company J, £2150.0, 75, 656734L
```

Summary

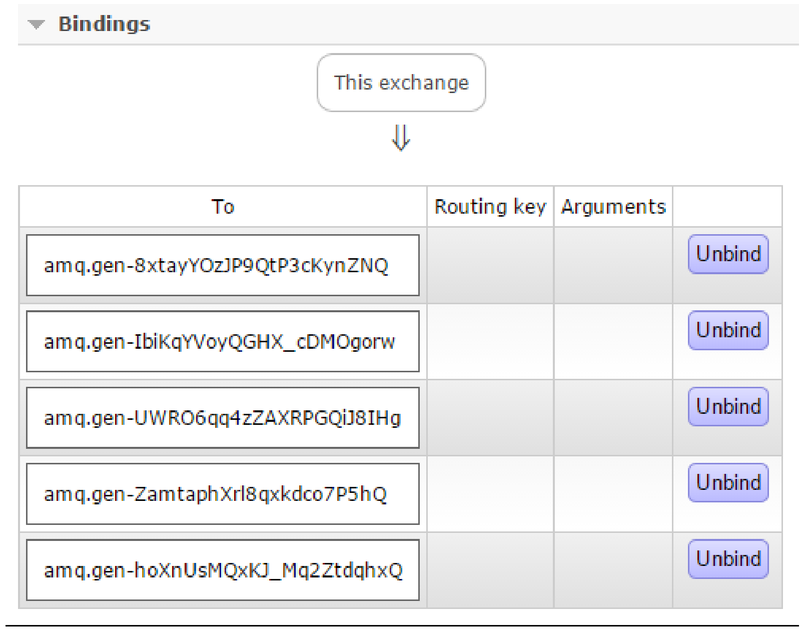




To	Routing key	Arguments	
CardPaymentDirectRouting_Queue	CardPayment		Unbind
PurchaseOrderDirectRouting_Queue	PurchaseOrder		Unbind


C:\Users\Stephen haunts\Dropbox (Personal)\Pluralsight\RabbitMQ By Example\rabbitmq-by-e...

```
-- Purchase Order - Routing Key <PurchaseOrder> : Company A, £50.0, 75, 123434A
-- Purchase Order - Routing Key <PurchaseOrder> : Company B, £150.0, 75, 193434B
-- Purchase Order - Routing Key <PurchaseOrder> : Company C, £12.0, 75, 196544A
-- Purchase Order - Routing Key <PurchaseOrder> : Company D, £2150.0, 75, 234434H
-- Purchase Order - Routing Key <PurchaseOrder> : Company E, £2150.0, 75, 876434W
-- Purchase Order - Routing Key <PurchaseOrder> : Company F, £7150.0, 75, 1423474U
-- Purchase Order - Routing Key <PurchaseOrder> : Company G, £3150.0, 75, 1932344O
-- Purchase Order - Routing Key <PurchaseOrder> : Company H, £3190.0, 75, 1123457Q
-- Purchase Order - Routing Key <PurchaseOrder> : Company I, £50.0, 75, 1595344R
-- Purchase Order - Routing Key <PurchaseOrder> : Company J, £2150.0, 75, 656734L
```



To	Routing key	Arguments	
amq.gen-8xtayYOzJP9QtP3cKynZNQ			Unbind
amq.gen-IbiKqYVoyQGHX_cDMOgorw			Unbind
amq.gen-UWRO6qq4zZAXRPGQiJ8IHg			Unbind
amq.gen-ZamtaphXrl8qxkdco7P5hQ			Unbind
amq.gen-hoXnUsMQxKJ_Mq2ZtdqhxQ			Unbind