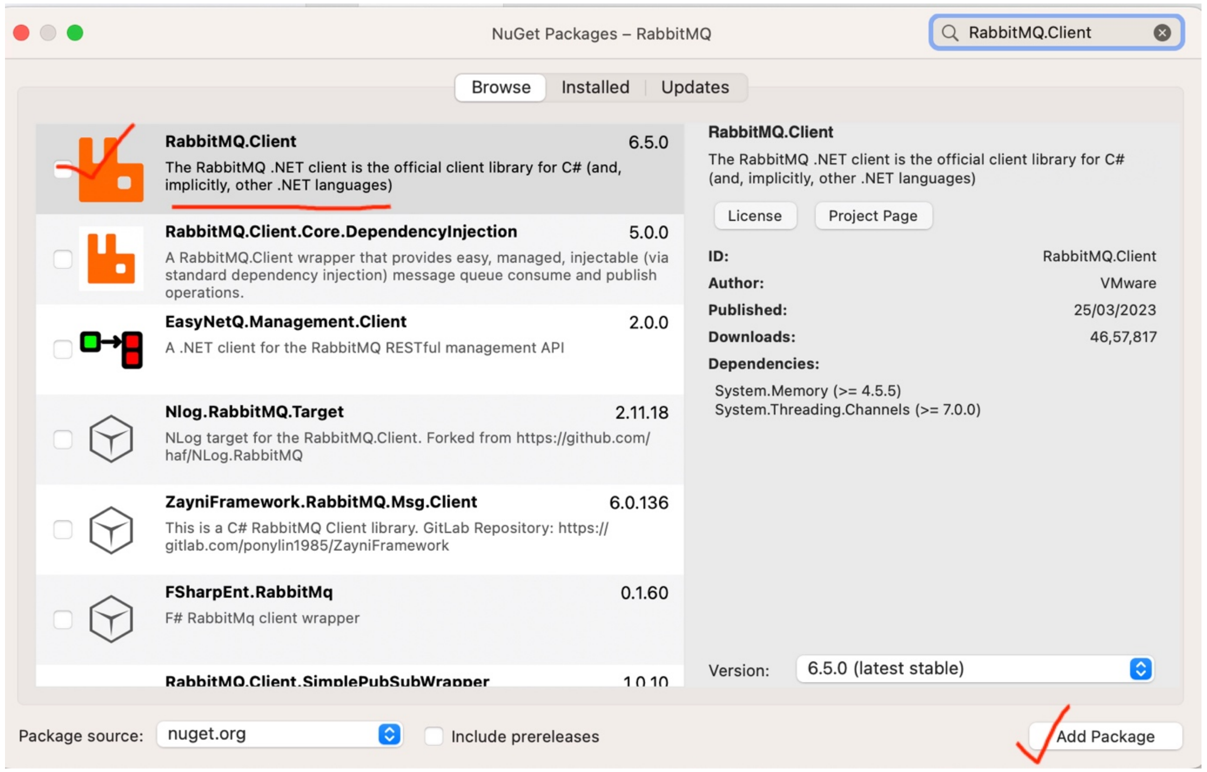
Table of Contents

1. Publishing and consuming RabbitMQ Message Using .NET – 60 Minutes

Create a .Net console application : RabbitMQ project.

Add the necessary .Net client using Nuget Package Explorer. (Manage Nuget Package)

RabbitMQ.Client --version 6.5.0



Package	Version	Description
RabbitMQ.Client	6.5.0	The RabbitMQ .NET client is the official client library for C# (and, implicitly, other .NET languages)
RabbitMQ.Client.Core.DependencyInjection	5.0.0	A RabbitMQ.Client wrapper that provides easy, managed, injectable (via standard dependency injection) message queue consume and publish operations.
EasyNetQ.Management.Client	2.0.0	A .NET client for the RabbitMQ RESTful management API
Nlog.RabbitMQ.Target	2.11.18	NLog target for the RabbitMQ.Client. Forked from https://github.com/haf/NLog.RabbitMQ
ZayniFramework.RabbitMQ.Msg.Client	6.0.136	This is a C# RabbitMQ Client library. GitLab Repository: https://gitlab.com/ponylin1985/ZayniFramework
FSharpEnt.RabbitMq	0.1.60	F# RabbitMq client wrapper
RabbitMQ.Client.SimplePubSubWrapper	1.0.10	

RabbitMQ.Client

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

ID:	RabbitMQ.Client
Author:	VMware
Published:	25/03/2023
Downloads:	46,57,817

Dependencies:

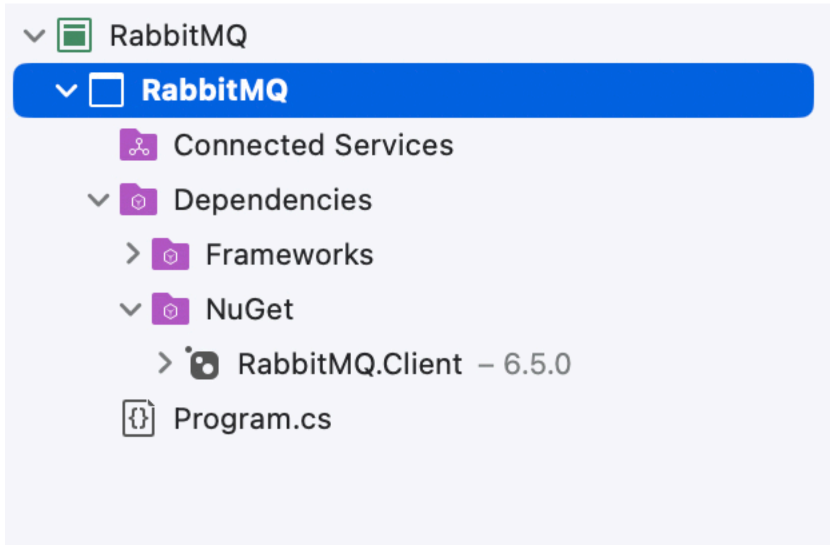
System.Memory (>= 4.5.5)
System.Threading.Channels (>= 7.0.0)

Version: 6.5.0 (latest stable)

Package source: nuget.org

Include prereleases

At the end you should have the Project view as shown below.



You should have the above packages.

Create a class SendingMessage.cs and add the following code in it.

It will push message to the RabbitMQ topic - nqueue.

```
using System;
using System;
using System.Text;
using RabbitMQ.Client;
```

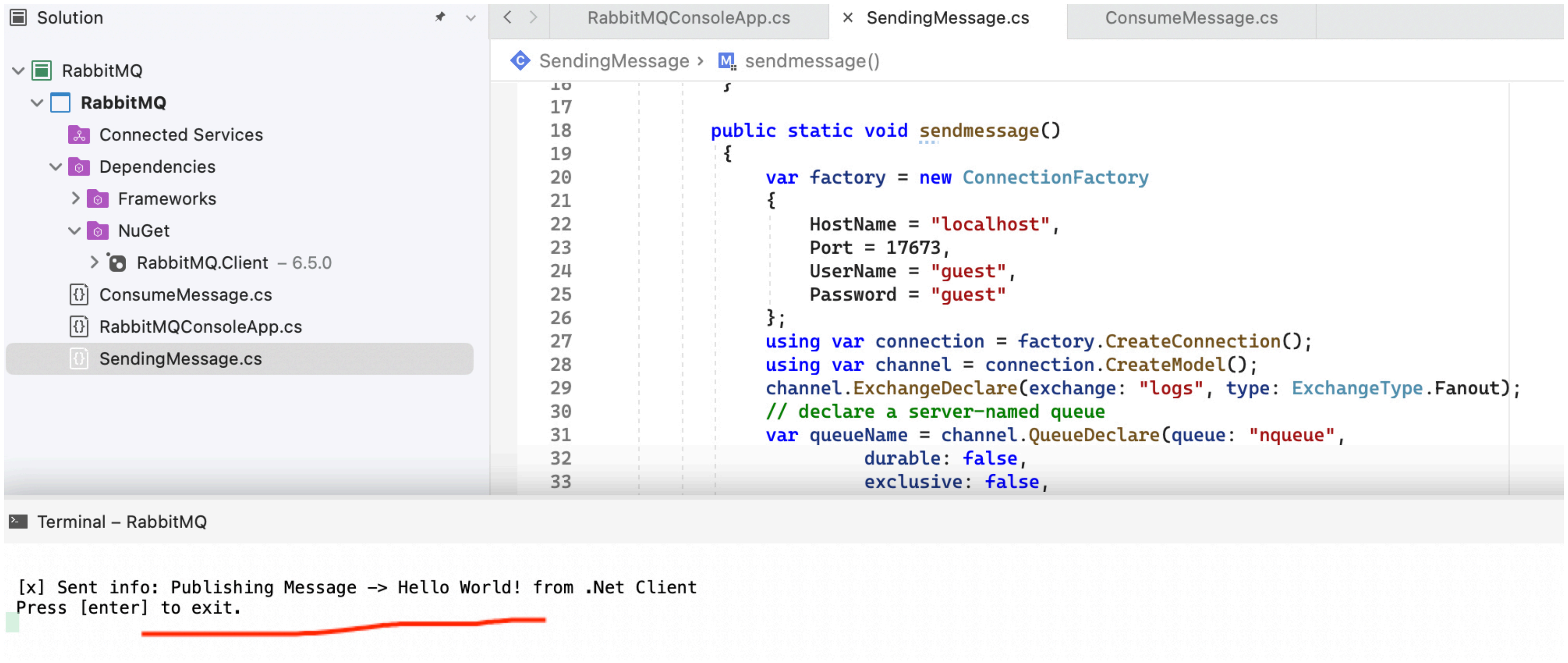
```
namespace RabbitMQ


{

    public class SendingMessage
        {
                public SendingMessage()
                {
                }

       public static void sendmessage()
        {
            var factory = new ConnectionFactory
            {
                HostName = "localhost",
                Port = 17673,
                UserName = "guest",
                Password = "guest"
            };
            using var connection = factory.CreateConnection();
            using var channel = connection.CreateModel();
            channel.ExchangeDeclare(exchange: "logs", type: ExchangeType.Fanout);
            // declare a server-named queue
            var queueName = channel.QueueDeclare(queue: "nqueue",
                    durable: false,
                exclusive: false,
                autoDelete: false,
                arguments: null);
            channel.QueueBind(queue: queueName,
                        exchange: "logs",
                        routingKey: string.Empty);
            var message = "info: Publishing Message -> Hello World! from .Net Client";
            var body = Encoding.UTF8.GetBytes(message);
            channel.BasicPublish(exchange: "logs",
                          routingKey: string.Empty,
                          basicProperties: null,
                          body: body);


            Console.WriteLine($" [x] Sent {message}");

            Console.WriteLine(" Press [enter] to exit.");
            Console.ReadLine();

        }


    }
}
```

Now, let us define a Main class that will invoke the above class and send message to topic.

RabbitMQConsoleApp.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace RabbitMQ
{
        public class RabbitMQConsoleApp
        {
                public RabbitMQConsoleApp()
                {
                }

      static void Main(string[] args)
      {
         SendingMessage.sendmessage();
         Console.WriteLine("Message Sent");
      }
   }
}
```

Execute the Program. You will get the following result.

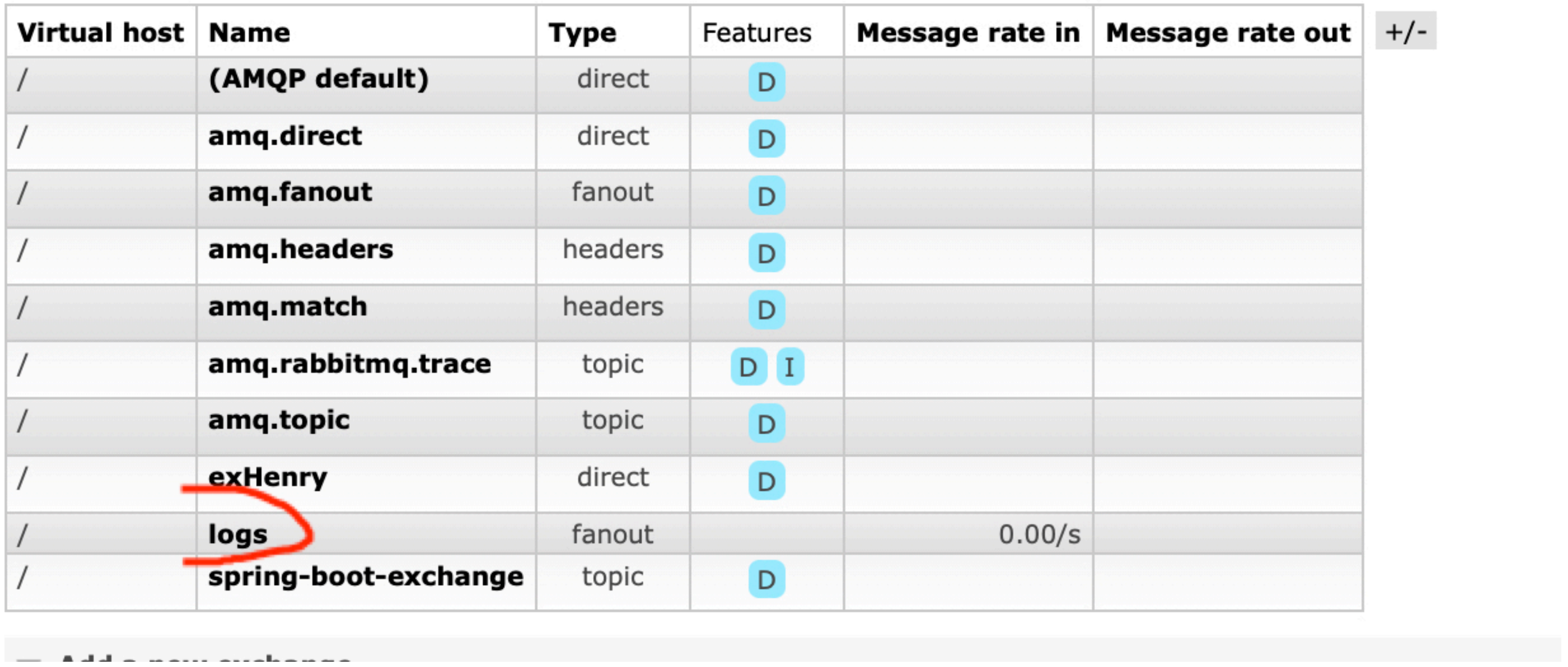


```
        public static void sendmessage()
        {
            var factory = new ConnectionFactory
            {
                HostName = "localhost",
                Port = 17673,
                UserName = "guest",
                Password = "guest"
            };
            using var connection = factory.CreateConnection();
            using var channel = connection.CreateModel();
            channel.ExchangeDeclare(exchange: "logs", type: ExchangeType.Fanout);
            // declare a server-named queue
            var queueName = channel.QueueDeclare(queue: "nqueue",
                     durable: false,
                     exclusive: false,
```

```
[x] Sent info: Publishing Message -> Hello World! from .Net Client
Press [enter] to exit.
```

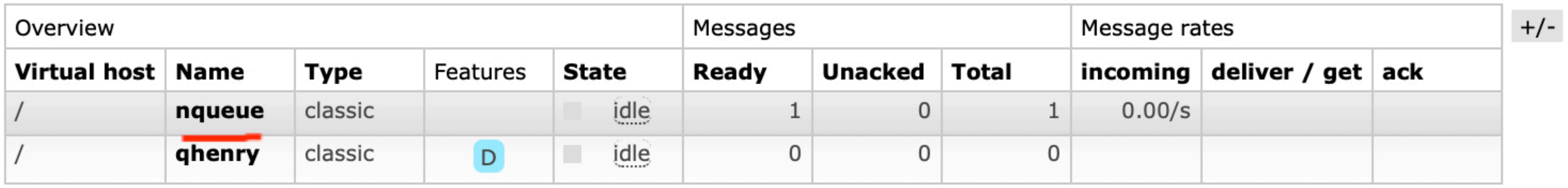
Let us verify from the web console.

Created exchange -> Logs



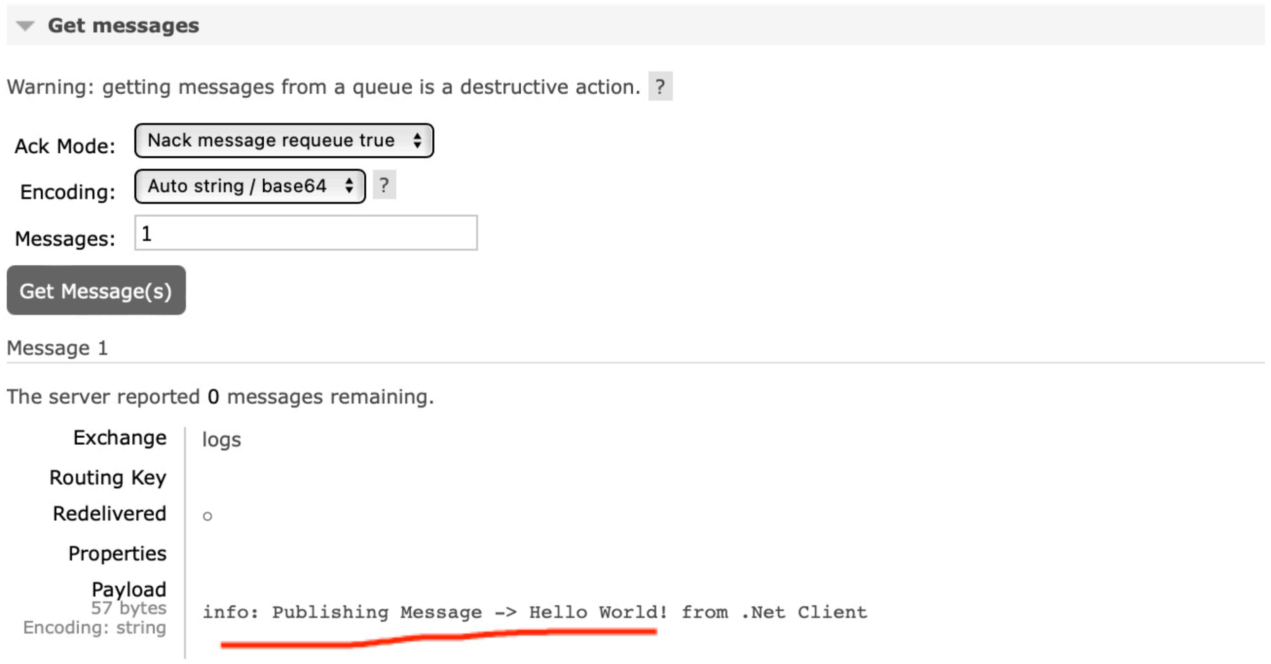
Virtual host	Name	Type	Features	Message rate in	Message rate out
/	(AMQP default)	direct	D		
/	amq.direct	direct	D		
/	amq.fanout	fanout	D		
/	amq.headers	headers	D		
/	amq.match	headers	D		
/	amq.rabbitmq.trace	topic	D I		
/	amq.topic	topic	D		
/	exHenry	direct	D		
/	logs	fanout		0.00/s	
/	spring-boot-exchange	topic	D		

And the message get store in the “nqueue” queue



Overview					Messages			Message rates		
Virtual host	**Name**	**Type**	Features	**State**	**Ready**	**Unacked**	**Total**	**incoming**	**deliver / get**	**ack**
/	**nqueue**	classic		idle	1	0	1	0.00/s		
/	**qhenry**	classic	D	idle	0	0	0			

You can verify the message from the console.



Warning: getting messages from a queue is a destructive action. ?

Ack Mode:	Nack message requeue true
Encoding:	Auto string / base64
Messages:	1

Message 1

The server reported **0** messages remaining.

Exchange	logs
Routing Key	
Redelivered	○
Properties	
Payload 57 bytes Encoding: string	`info: Publishing Message -> Hello World! from .Net Client`

Now let us consume the message using .Net API.

Create a class and Add the following method - consumeMessage().

Class Name : ConsumeMessage.cs

```
using System;
using System.Text;
using RabbitMQ.Client;
using RabbitMQ.Client.Events;

namespace RabbitMQ
{
        public class ConsumeMessage
        {
                public ConsumeMessage()
                {
                }

                public static void consumeMessage() {
         var factory = new ConnectionFactory
         {
            HostName = "localhost",
            Port = 17673,
            UserName = "guest",
            Password = "guest"
         };
         using var connection = factory.CreateConnection();
         using var channel = connection.CreateModel();
```

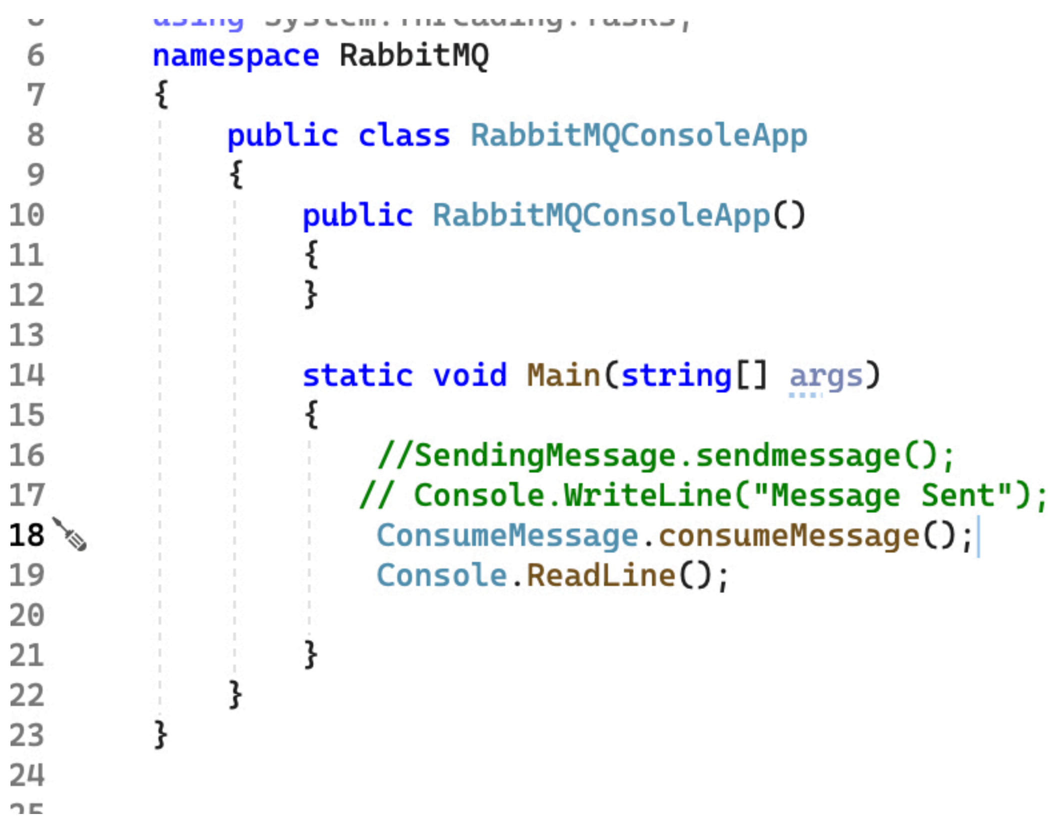
```
channel.ExchangeDeclare(exchange: "logs", type: ExchangeType.Fanout);

// declare a server-named queue
var queueName =  channel.QueueDeclare(queue: "nqueue",
        durable: false,
        exclusive: false,
        autoDelete: false,
        arguments: null);

channel.QueueBind(queue: queueName,
                exchange: "logs",
                routingKey: string.Empty);

Console.WriteLine(" [*] Waiting for logs.");

var consumer = new EventingBasicConsumer(channel);
consumer.Received += (model, ea) =>
{
   byte[] body = ea.Body.ToArray();
   var message = Encoding.UTF8.GetString(body);
   Console.WriteLine($" [x] {message}");
};
channel.BasicConsume(queue: queueName,
                  autoAck: true,
                  consumer: consumer);

Console.WriteLine(" Press [enter] to exit.");
```

```
            Console.ReadLine();
        }
          }
}
```

Update the RabbitMQConsoleApp.cs to invoke the consumer method. You need to comment the sender method.

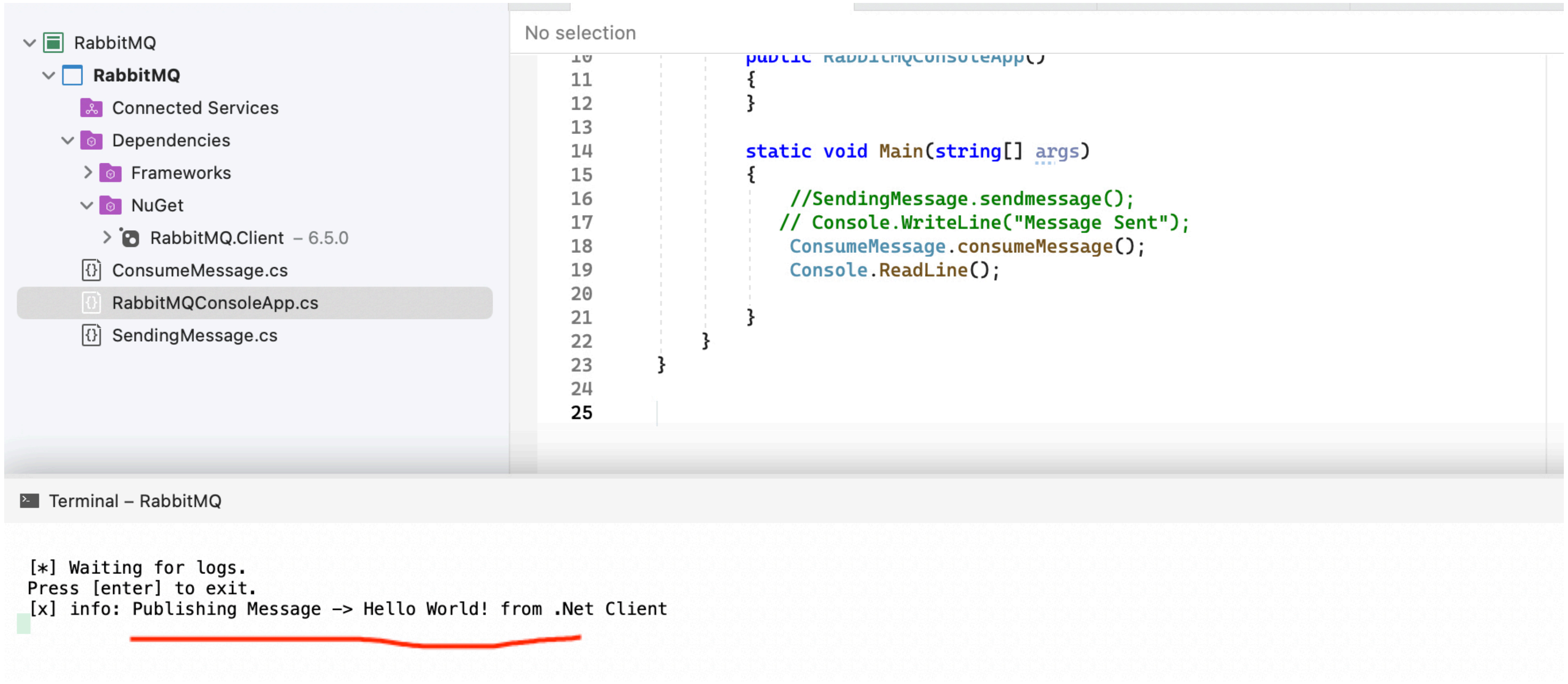
```
           //SendingMessage.sendmessage();
           // Console.WriteLine("Message Sent");
            ConsumeMessage.consumeMessage();
            Console.ReadLine();
```

Your main program should like as shown below



```
using System.Threading.Tasks;
namespace RabbitMQ
{
    public class RabbitMQConsoleApp
    {
        public RabbitMQConsoleApp()
        {
        }

        static void Main(string[] args)
        {
            //SendingMessage.sendmessage();
           // Console.WriteLine("Message Sent");
            ConsumeMessage.consumeMessage();
            Console.ReadLine();

        }
    }
}
```

Execute the main program. You should be able to see the following message.



```
        public RabbitMQConsoleApp()
        {
        }

        static void Main(string[] args)
        {
            //SendingMessage.sendmessage();
           // Console.WriteLine("Message Sent");
            ConsumeMessage.consumeMessage();
            Console.ReadLine();

        }
    }
}

```

```
[*] Waiting for logs.
Press [enter] to exit.
[x] info: Publishing Message -> Hello World! from .Net Client
```

------- --- Lab Ends Here --