**RabbitMQ 流程**

1.安装otp\_win64\_23.2 ---语言

2. rabbitmq-server-3.8.14 ---服务

重启rabbitmq服务通过两个命令来实现：  
rabbitmqctl stop ：停止rabbitmq  
rabbitmq-server restart : 重启rabbitmq

地址：https://github.com/sheng-jie/RabbitMQ

<https://www.rabbitmq.com/tutorials/tutorial-five-dotnet.html>

服务：

本文主要基于Windows下使用Vs Code 基于.net core进行demo演示。开始之前我们需要准备好以下环境。

安装Erlang运行环境 下载安装[Erlang](http://www.erlang.org/downloads)。

安装RabbitMQ 下载安装Windows版本的[RabbitMQ](http://www.rabbitmq.com/install-windows.html)。

启动RabbitMQ Server 点击Windows开始按钮，输入RabbitMQ找到RabbitMQ Comman Prompt，以管理员身份运行。

依次执行以下命令启动RabbitMQ服务

rabbitmq-service install

rabbitmq-service enable

rabbitmq-service start

执行rabbitmqlctl status检查RabbitMQ状态

安装管理平台插件 执行rabbitmq-plugins enable rabbitmq\_management即可成功安装，使用默认账号密码（guest/guest）登录<http://localhost:15672/>

创建RabbitMQ文件夹，打开命令提示符，分别创建两个控制台项目Send、Receive。

dotnet new console --name Send //创建发送端控制台应用

cd Send //进入Send目录

dotnet add package RabbitMQ.Client //添加RabbitMQ.Client包

dotnet restore //恢复包

dotnet new console --name Receive //创建接收端控制台应用

cd Receive //进入Receive目录

dotnet add package RabbitMQ.Client //添加RabbitMQ.Client包

dotnet restore //恢复包

direct（明确的路由规则：消费端绑定的队列名称必须和消息发布时指定的路由名称一致）

topic （模式匹配的路由规则：支持通配符）

fanout （消息广播，将消息分发到exchange上绑定的所有队列上）

它只是将接收到的所有消息广播到它知道的所有队列中。而这正是我们记录器所需要的。

**RabbitMQ，RocketMQ 和 Kafka 几种常见的消息队列主要特点和优劣势。**

本文分别介绍了 RabbitMQ，RocketMQ 和 Kafka 几种常见的消息队列，阐述了各种消息队列的主要特点和优劣势。

在了解了上面这些开源消息队列各自的特点和优劣势后，对于消息队列及相关技术选型，相信你会有更深入的理解和认识。以下几条选择的建议可以参考：

如果消息队列不是将要构建系统的重点，对消息队列功能和性能没有很高的要求，只需要一个快速上手易于维护的消息队列，建议使用 RabbitMQ。

如果系统使用消息队列主要场景是处理在线业务，比如在交易系统中用消息队列传递订单，需要低延迟和高稳定性，建议使用 RocketMQ。

如果需要处理海量的消息，像收集日志、监控信息或是埋点这类数据，或是你的应用场景大量使用了大数据、流计算相关的开源产品，那 Kafka 是最适合的消息队列。

每一个消息队列都有自己的优劣势，需要根据现有系统的情况，选择最适合的消息队列，更多细节和原理性的东西，还需在实践中见真知！

**本文分别介绍了 RabbitMQ，RocketMQ 和 Kafka 几种常见的消息队列，阐述了各种消息队列的主要特点和优劣势。**

在了解了上面这些开源消息队列各自的特点和优劣势后，对于消息队列及相关技术选型，相信你会有更深入的理解和认识。以下几条选择的建议可以参考：

•如果消息队列不是将要构建系统的重点，对消息队列功能和性能没有很高的要求，只需要一个快速上手易于维护的消息队列，建议使用 RabbitMQ。•如果系统使用消息队列主要场景是处理在线业务，比如在交易系统中用消息队列传递订单，需要低延迟和高稳定性，建议使用 RocketMQ。•如果需要处理海量的消息，像收集日志、监控信息或是埋点这类数据，或是你的应用场景大量使用了大数据、流计算相关的开源产品，那 Kafka 是最适合的消息队列。

每一个消息队列都有自己的优劣势，需要根据现有系统的情况，选择最适合的消息队列，更多细节和原理性的东西，还需在实践中见真知！



基于上面的demo和对几种不同exchange路由机制的学习，我们发现RabbitMQ主要是涉及到以下几个核心概念：

1. Publisher：生产者，消息的发送方。
2. Connection：网络连接。
3. Channel：信道，多路复用连接中的一条独立的双向数据流通道。
4. Exchange：交换器（路由器），负责消息的路由到相应队列。
5. Binding：队列与交换器间的关联绑定。消费者将关注的队列绑定到指定交换器上，以便Exchange能准确分发消息到指定队列。
6. Queue：队列，消息的缓冲存储区。
7. Virtual Host：虚拟主机，虚拟主机提供资源的逻辑分组和分离。包含连接，交换，队列，绑定，用户权限，策略等。
8. Broker：消息队列的服务器实体。
9. Consumer：消费者，消息的接收方。

这次作为入门就讲到这里，下次我们来讲解下**EventBus + RabbitMQ**如何实现事件的分发。

案例：

Dome: https://github.com/sheng-jie/RabbitMQ