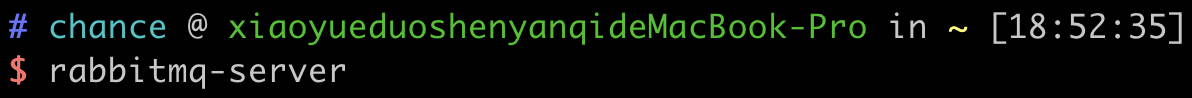
RabbitMQ[安装指南](https://www.rabbitmq.com/download.html)

前台运行：



登陆管理界面：<http://localhost:15672>

账号密码初始默认密码：guest

本教程涵盖使用RabbitMQ创建消息传递应用程序的基础知识。

### 1、HelloWorld

The simplest thing that does something。



#### Introduction

RabbitMQ is a message broker: it accepts and forwards messages. You can think it as a post office: when you put the mail that you want posting in a post box, you can be sure that Mr. or Ms. Mailperson will eventually deliver the mail to your recipient. In this analogy, RabbitMQ is a post box, a post office and a postman. （RabbitMQ是消息代理：它接受并转发消息。 您可以将其视为邮局：将要发布的邮件放在邮箱中时，可以确保Mailperson先生或女士最终将邮件传递给收件人。 以此类推，RabbitMQ是一个邮箱，一个邮局和一个邮递员。）

The major difference between RabbitMQ and the post office is that it doesn't deal with paper, instead it accepts, stores and forwards binary blobs of data ‒ messages. （RabbitMQ与邮局之间的主要区别在于，它不处理纸张，而是接收，存储和转发数据-消息的二进制Blob。）

RabbitMQ, and messaging in general, uses some jargon.

* Producing means nothing more than sending. A program that sends messages is a producer: （生产仅仅意味着发送。发送消息的程序是生产者）



* A queue is the name for a post box which lives inside RabbitMQ. Although messages flow through RabbitMQ and your applications, they can only be stored inside a queue. A queue is only bound by the host's memory & disk limits, it's essentially a large message buffer. Many producers can send messages that go to one queue, and many consumers can try to receive data from one queue. This is how we represent a queue: （队列是RabbitMQ内部的邮箱的名称。尽管消息流经RabbitMQ和你的应用，它们只可以被存储在队列中。队列仅受主机内存和磁盘的限制，本质上是一个大的消息缓冲区。许多生产者可以将消息发送到一个队列，许多消费者可以尝试从一个队列接收数据。这就是我们表示队列的方式）



* Consuming has a similar meaning to receiving. A consumer is a program that mostly waits to receive messages: （消费与接收具有相似的含义。消费者是一个主要等待接收消息的程序。）



Note that the producer, consumer, and broker do not have to reside on the same

host; indeed in most applications they don't. An application can be both a producer and consumer, too. （请注意，生产者，消费者和队列不必位于同一主机上。实际上，大多数应用程序中它们都不要。一个应用程序既可以是生产者又可以是消费者。）

"Hello World"

#### (using the Java Client)

In this part of the tutorial we'll write two programs in Java; a producer that sends a single message, and a consumer that receives messages and prints them out. We'll gloss over some of the detail in the Java API, concentrating on this very simple thing just to get started. It's a "Hello World" of messaging. （在本教程的这一部分中，我们将用Java编写两个程序。发送单个消息的生产者和接收消息并打印出来的消费者。我们将介绍Java API中的一些细节，仅着眼于此非常简单的事情。这是消息传递的“ Hello World”。）

In the diagram below, "P" is our producer and "C" is our consumer. The box in the middle is a queue - a message buffer that RabbitMQ keeps on behalf of the consumer. （在下图中，“ P”是我们的生产者，“ C”是我们的消费者。中间的框是一个队列-RabbitMQ代表使用者保留的消息缓冲区。）



**The Java client library（Java客户端库）**

RabbitMQ speaks multiple protocols. This tutorial uses AMQP 0-9-1, which is an open, general-purpose protocol for messaging. There are a number of clients for RabbitMQ in many different languages. We'll use the Java client provided by RabbitMQ. （RabbitMQ使用多种协议。 本教程使用AMQP 0-9-1，这是一种开放的通用消息传递协议。 RabbitMQ有许多不同语言的客户。 我们将使用RabbitMQ提供的Java客户端。）

Download the client library and its dependencies (SLF4J API and SLF4J Simple). Copy those files in your working directory, along the tutorials Java files. （下载客户端库及其依赖项（SLF4J API和SLF4J Simple）。 将这些文件和教程Java文件一起复制到您的工作目录中。）

Please note SLF4J Simple is enough for tutorials but you should use a full-blown logging library like Logback in production. （请注意，对于教程而言，SLF4J Simple足够了，但是您应该在生产中使用成熟的日志记录库，例如Logback。）

(The RabbitMQ Java client is also in the central Maven repository, with the groupId com.rabbitmq and the artifactId amqp-client.) （RabbitMQ Java客户端也位于中央Maven存储库中，带有groupId com.rabbitmq和artifactId amqp-client。）

Now we have the Java client and its dependencies, we can write some code.

#### Sending



We'll call our message publisher (sender) Send and our message consumer (receiver) Recv. The publisher will connect to RabbitMQ, send a single message, then exit. （我们将其称为消息发布者（发送者）Send，将消息消费者（接收者）Recv。 发布者将连接到RabbitMQ，发送一条消息，然后退出。）

创建Send.java类

2、Work queues

3、Publlish/Subscribe

4、Routing

5、Topics

6、RPC

7、Publisher Confirms