

一. RabbitMQ

要实现工厂自动化，可靠的信息传递是必要的

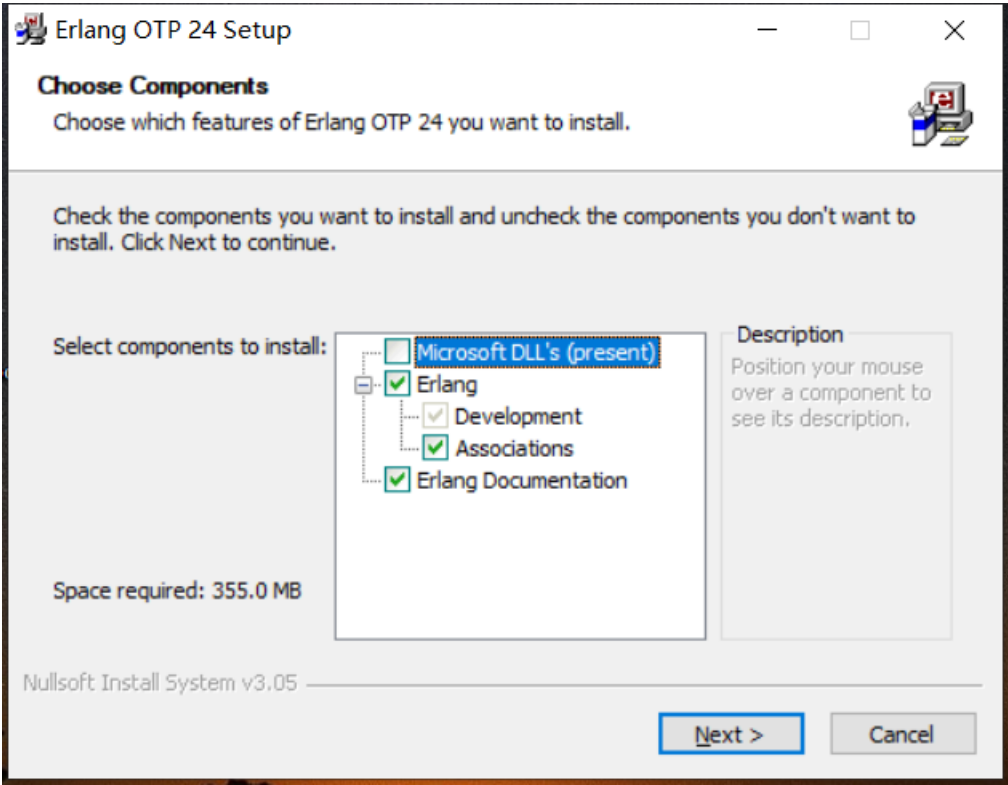
RabbitMQ 是高级消息队列协议(AMQP)的开源消息代理软件。RabbitMQ 服务器是用 Erlang 语言编写的，消息系统允许软件、应用相互连接和扩展。这些应用可以相互链接起来组成一个更大的应用， 或者将用户设备和数据进行连接。消息系统通过将消息的发送和接收分离来实现应用程序的异步和解偶。或许你正在考虑进行数据投递，非阻塞操作或推送通知。或许你想要实现发布 / 订阅，异步处理， 或者工作队列。所有这些都可以通过消息实现。RabbitMQ 是一个消息代理 - 一个消息系统的媒介。它可以为你的应用提供一个通用的消息发送和接收平台，并且保证消息在传输过程中的安全。

二. 安装 RabbitMQ

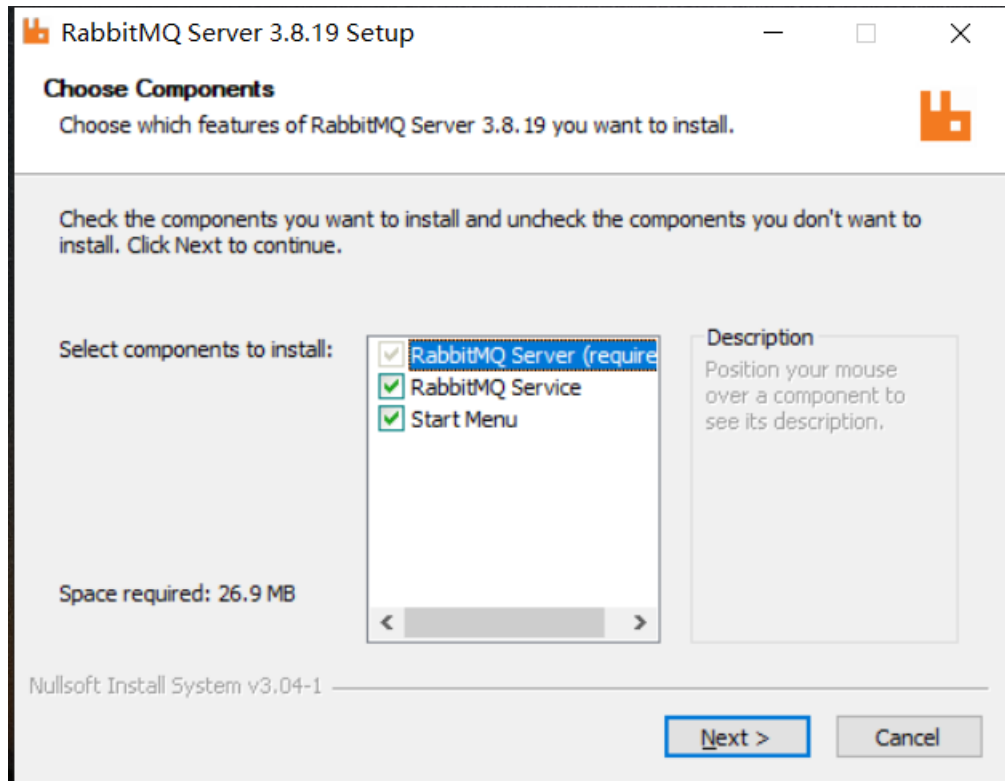
RabbitMQ 需要安装 64 位支持的 Erlang for Windows 版本
所需框架与安装版本官网均可免费下载

名称	修改日期	类型	大小
今天 (2)			
otp_win64_24.0.exe	2021/7/22 22:30	应用程序	112,696 KB
rabbitmq-server-3.8.19.exe	2021/7/22 20:00	应用程序	17,469 KB

1 安装 otp_win64_24.0.exe (Erlang) 因为 RabbitMQ 它依赖于 Erlang,需要先安装 Erlang



2. 安装 RabbitMQ Server



3. 安装完成后 打开命令提示符 cd 到 rabbitmq_server-3.8.19\sbin 目录下

输入命令:

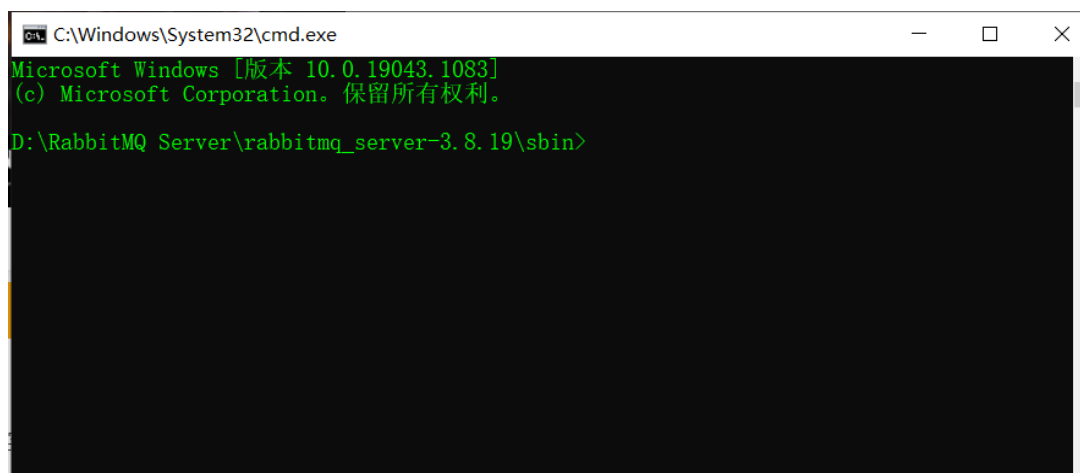
rabbitmq-plugins enable rabbitmq_management

停止: net stop RabbitMQ

启动: net start RabbitMQ

在浏览器中输入地址查看: <http://127.0.0.1:15672/>

使用默认账号登录: guest/ guest



1.写个生产消息 Demo


1.写个生产消息 Demo

```
//rabbitmqctl stop_service 停止服务,请清除queue
//rabbitmqctl start_service 开始服务
//rabbitmqctl list_queues 查询当前队列
//rabbitmqctl purge_queue kibaQueue 清空指定queue队列的数据
0 个引用
class Program
{
    0 个引用
    static void Main(string[] args)
    {
        var factory = new ConnectionFactory();
        factory.HostName = "localhost";//主机名, Rabbit会拿这个IP生成一个endpoint, 这个很熟悉吧, 就是socket绑定的那个终结点。
        factory.UserName = "guest";//默认用户名, 用户可以在服务端自定义创建, 有相关命令行
        factory.Password = "guest";//默认密码

        using (var connection = factory.CreateConnection())//连接服务器, 即正在创建终结点。
        {
            //创建一个通道, 这个就是Rabbit自己定义的规则了, 如果自己写消息队列, 这个就可以开始搞设计了
            //这里Rabbit的玩法就是一个通道channel下包含多个队列Queue
            using (var channel = connection.CreateModel())
            {
                for (int i = 0; i < 100; i++)
                {
                    channel.QueueDeclare("TestQueue", false, false, false, null);//创建一个名称为TestQueue的消息队列
                    var properties = channel.CreateBasicProperties();
                    properties.DeliveryMode = 1;
                    string message = "传递的消息内容 " + i; //传递的消息内容
                    channel.BasicPublish("", "TestQueue", properties, Encoding.UTF8.GetBytes(message)); //生产消息
                    Console.WriteLine($"Send: {message}");
                    Thread.Sleep(3000);
                }
            }
        }
        Console.ReadLine();
    }
}
```

[illegible]

2.登录 Rabbit 服务可以看到生产的消息


Refreshed 2021-07-24 15:56:53
Refresh every 5 seconds

RabbitMQ 3.8.19
Erlang 24.0

Virtual host All
Cluster rabbit@DESKTOP-AN0UR35
User guest
Log out

Overview
Connections
Channels
Exchanges
Queues

Admin

Queues

▼ All queues (1)

Page 1 of 1
Filter:
Regexp ?
Displaying 1 item, page size up to: 100

Overview					Messages			Message rates		
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
/	TestQueue	classic		running	21	0	21	0.20/s	0.00/s	0.00/s

► Add a new queue

HTTP API
Server Docs
Tutorials
Community Support
Community Slack
Commercial Support
Plugins
GitHub
Changelog

3.再写个接受消息 Demo

```

0 个引用
static void Main(string[] args)
{
    var factory = new ConnectionFactory();
    factory.HostName = "localhost";
    factory.UserName = "guest";
    factory.Password = "guest";

    using (var connection = factory.CreateConnection())
    {
        using (var channel = connection.CreateModel())
        {
            channel.QueueDeclare("TestQueue", false, false, false, null);

            /* 这里定义了一个消费者，用于消费服务器接受的消息
             * 这里，其实就是定义一个EventingBasicConsumer类型的对象，然后该对象有个Received事件，该事件会在服务接收到数据
             */
            var consumer = new EventingBasicConsumer(channel); //消费者
            channel.BasicConsume("TestQueue", true, consumer); //消费消息 autoAck参数为消费后是否删除
            consumer.Received += (model, ea) =>
            {
                var body = ea.Body;
                var message = Encoding.UTF8.GetString(body);
                Console.WriteLine("Received: {0}", message);
            };
            Console.ReadLine();
        }
    }
}
}

```

[illegible]