

**ARTL for Java**

Java Technical Good

Introduction to Apache Camel

Install ActiveMQ on windows

*Quick Start and User Guide*

*June 2015 version 1.00*

*March, 2013*

*Version 1.0*

Table of Contents

[1 Apache Camel 3](#_Toc350505671)

[1.1 Introduction 3](#_Toc350505672)

[1.2 The Problem : Enterprise Application Integration (EAI) 3](#_Toc350505673)

[1.3 Enterprise Integration Patterns (EIP) 3](#_Toc350505674)

[1.4 What is Apache Camel? 4](#_Toc350505673)

[1.5 When to use Apache Camel ? (features) 4](#_Toc350505674)

[1.6 When NOT to use Apache Camel ? 5](#_Toc350505673)

[1.7 Conclusion 5](#_Toc350505674)

[2 Installing ActiveMQ on windows 5](#_Toc350505685)

[2.1 Download ActiveMQ 5](#_Toc350505686)

[2.2 Review ActiveMQ folder 6](#_Toc350505687)

[2.3 Run ActiveMQ server 6](#_Toc350505686)

[2.4 Open ActiveMQ admin console in browser 7](#_Toc350505687)

[4 Document Control 8](#_Toc350505689)

[4.1 Change History 8](#_Toc350505690)

[4.2 Open Issues 8](#_Toc350505691)

# Apache Camel

## Introduction

Apache Camel is one of the open source frameworks in the JVM / Java environment. It enables easy integration of different applications which use several protocols and technologies.

## The Problem: Enterprise Application Integration (EAI)

Enterprise application integration is necessary in almost every company due to new products and applications. Integrating these applications creates several problems. New paradigms come up every decade, for example client / server communication, Service-oriented Architecture (SOA) or Cloud Computing.

Besides, different interfaces, protocols and technologies emerge. Instead of storing data in files in the past (many years ago), SQL databases are used often today. Sometimes, even NoSQL databases are required in some use cases. Synchronous remote procedure calls or asynchronous messaging is used to communicate via several technologies such as RMI, SOAP Web Services, REST or JMS. A lot of software silos exists. Nevertheless, all applications and products of these decades have to communicate with each other to work together perfectly

## Enterprise Integration Patterns (EIP)

Enterprise Integration Patterns (www.eaipatterns.com) help to fragment problems and use standardized ways to integrate applications. Using these, you always use the same concepts to transform and route messages.

Three alternatives exist for integrating applications. EIPs can be used in each solution.

• ***Solution 1: Own custom Solution***

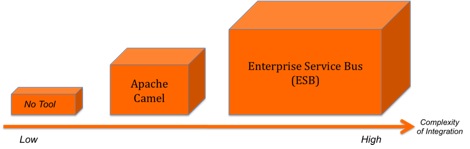
Implement an individual solution that works for your problem without separating problems into little pieces. This works and is probably the fastest alternative for small use cases. Lots of coding has to be done. Maintenance will probably be high if team members change.

• ***Solution 2: Integration Framework***

Use a framework which helps to integrate applications in a standardized way using several integration patterns. It reduces efforts a lot. Every developer will easily understand what is done (if he knows the used framework).

• ***Solution 3: Enterprise Service Bus (ESB)***

Use an enterprise service bus to integrate applications. Under the hood, the ESB also uses an integration framework. But there is much more functionality, such as business process management, a registry or business activity monitoring. One can usually configure routing and such stuff within a graphical user interface – you have to decide at your own if that reduces complexity and efforts. Usually, an ESB is a complex product. The learning curve is much higher.



## What is Apache Camel?

## Apache Camel is a lightweight integration framework which implements all EIPs.

## Easy integration of different applications using the required patterns.

## Camel empowers you to define routing and mediation rules in a variety of domain-specific languages, including a Java-based Fluent API, Spring or Blueprint XML Configuration files, and a Scala DSL.

## Almost every technology is available, for example HTTP, FTP, JMS, EJB, JPA, RMI, JMS, JMX, LDAP, Netty, and many, many more (of course most ESBs also offer support for them). Besides, own custom components can be created very easily.

* You can deploy Apache Camel as standalone application, in a web container (e.g. Tomcat or Jetty), in a JEE application Server (e.g. JBoss AS or WebSphere AS), in an OSGi environment or in combination with a Spring container.

## When to use Apache Camel? (features)

* Apache Camel is awesome if you want to integrate several applications with different protocols and technologies. Why? There is one feature (besides supporting so many technologies and besides supporting different programming languages) which is most attractive : Every integration uses the same concepts! No matter which protocol you use. No matter which technology you use. No matter which domain specific language (DSL) you use – it can be Java, Scala, Groovy or Spring XML. You do it the same way. Always! There is a producer, there is a consumer, there are endpoints, there are EIPs, there are custom processors / beans (e.g. for custom transformation) and there are parameters (e.g. for credentials).
* Two other very important features are the support for error-handling (e.g. using a dead letter queue) and automatic testing. You can test EVERYTHING very easily using a Camel-extension of JUnit! And again, you always use the same concepts, no matter which technology you have to support.
* Apache Camel is mature and production ready. It offers scalability, transaction support, concurrency and monitoring.

## When NOT to use Apache Camel?

* If there is just one or two technologies for integration , e.g. reading a file or sending a JMS message, it is probably much easier and faster to use some well-known libraries such as Apache Commons IO or Spring JmsTemplate.
* For very large integration projects camel might not be a right choice. An ESB is the right tool for this job in most cases. It offers many additional features such as BPM or BAM. Well-known open source ESBs are Apache ServiceMix, Mule ESB and WSO2 ESB.
* Some ESB base on the Apache Camel framework (e.g. Apache Service Mix and the Talend ESB).

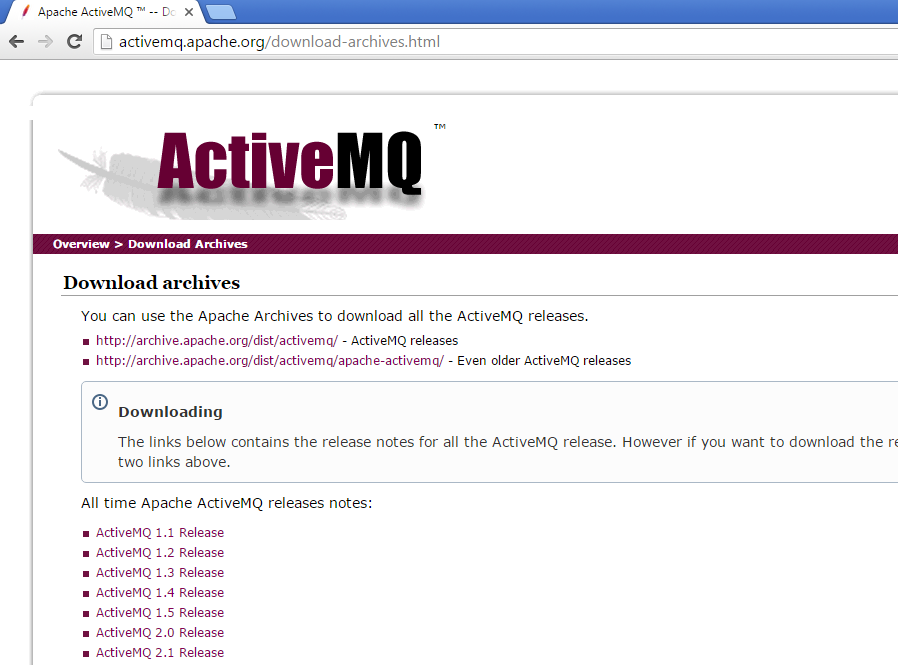
## Conclusion

Apache Camel is an awesome framework to integrate applications with different technologies. The best thing is that you always use the same concepts. Besides, support for many technologies, good error handling and easy automatic testing make it ready for integration projects.

# installing mongodb on windows

## Download ActiveMQ

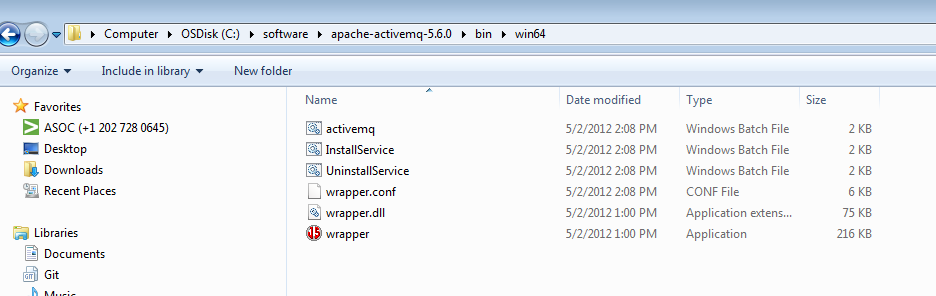
Download ActiveMQ from official ActiveMQ website. Choose the windows distribution. Unzip, extracts to your prefer location, for example : C:\software\apache-activemq-5.6.0



## Review ActiveMQ folder

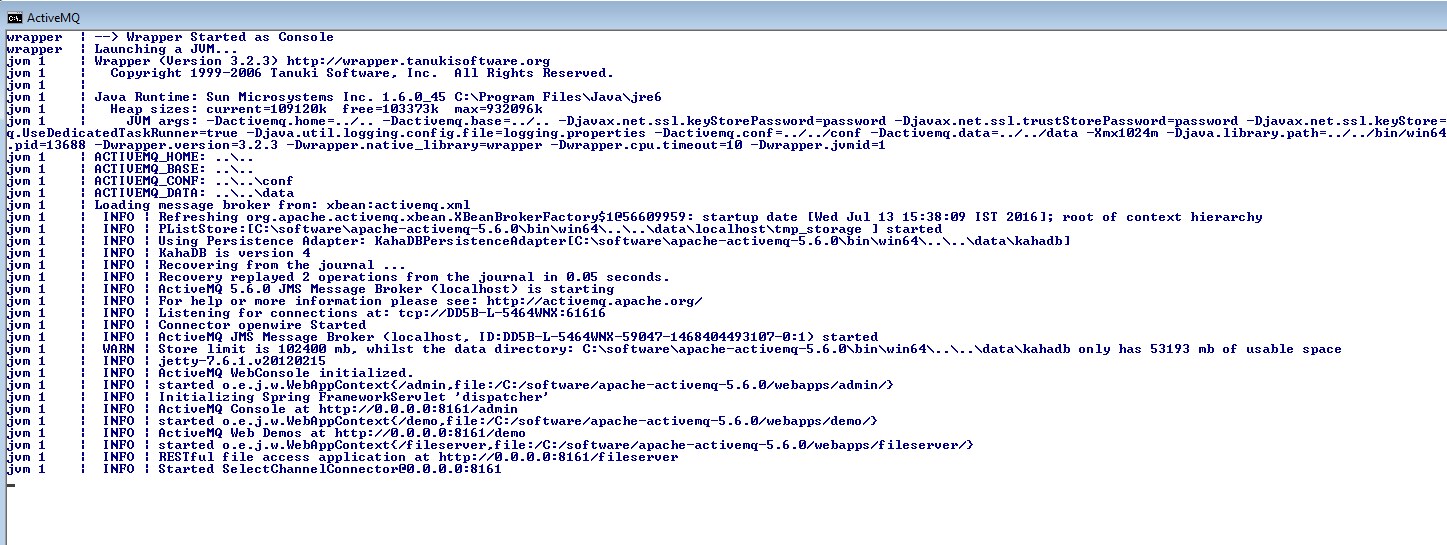
Go to the bin folder of the extracted ActiveMQ folder. You will find win32 and win64 folder. Chose according to your system. It will look like the below screenshot.

Figure : Files under C:\software\apache-activemq-5.6.0\bin\win64



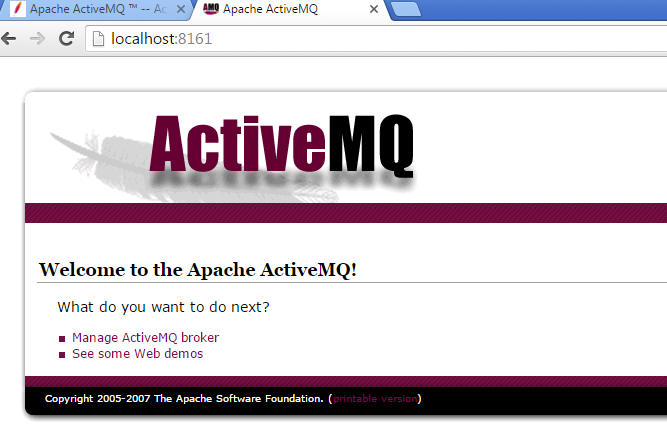
## Run ActiveMQ server

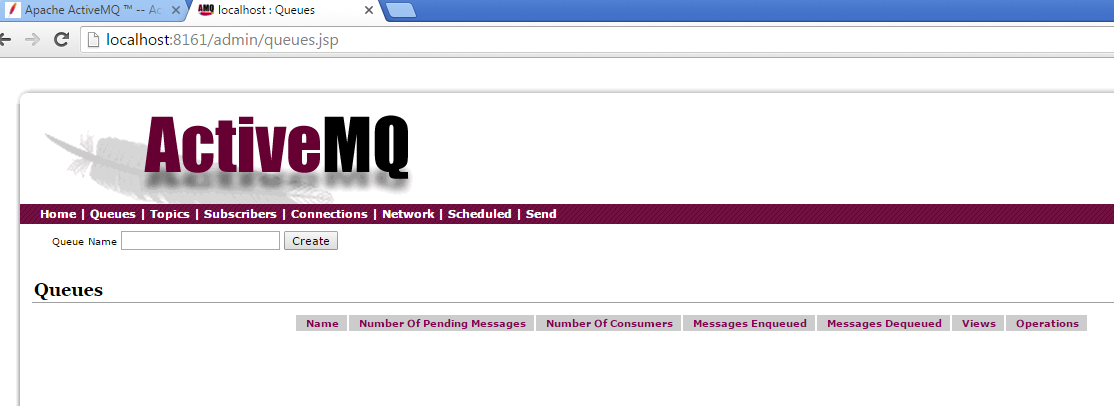
Double click on activemq batch file which is shown in the screen shot as above and that’s it. It will start the ActiveMQ server . Refer below careen shot for better understanding.



## 2.4 Open ActiveMQ level in the console

Open default url <http://localhost:8161/> in our browser to verify the ActiveMQ server.





# 

# Document Control

## Change History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Approver** | **Comment** |
| 1 | 13/07/2016 | 1 | Ashish Agarwal |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Open Issues

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Action** | **Owner** | **Status** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |