**What’s JMS ?** In short, JMS is a set of APIs defined in JSR. In practical you can think it’s a set of Java Interfaces. JMS is part of the JEE standard coming with JDK in package***javax.jms.\****  There are basically 2 versions of JMS, JMS 1.x and JMS 2.0. These 2 sets of APIs are different.   
**What’s ActiveMQ?** In short ActiveMQ is one of the implementation of JMS APIs (Of course there are other implementations, Open JMS, RabbitMQ for example).  ActiveMQ also provide the broker which can be treated somehow  like a server, that you can send message to or receive message from. For now, 2014-10, ActiveMQ only implements the interfaces defined in  JMS 1.1 version.    
**What’s Spring JMS?** Spring JMS is part of  the whole spring framework. It wrap the real JMS service provider such as ActiveMQ or OpenJMS, provides consistent APIs to upper logic. Spring  JMS’s APIs are quite similar to JMS2.0  in JEE7. Spring JMS can decouple you business logic from the real JMS service provider, which here in our example is apache ActiveMQ.   
In this example we are going to create a example to send out a text message by using Spring JMS and ActiveMQ.

**1. What you need**

* JDK 1.7
* Maven 3.2.1
* ActiveMQ 5.10.0
* Spring 4.1.0.RELEASE  (acquired by Maven)

We’ll use maven to manage all dependencies.  To write code there is no need for ActiveMQ binary, since maven will take care of the ActiveMQ library we need. But to run the code, we need the ActiveMQ binary, In this example  we'll run the ActiveMQ broker on a machine of IP 192.168.203.143 with default port 61616. If you run the broker in a different IP or port, don't forget to change the broker URL in Spring configuration file.