**JMS** is a standard Java API that allows a Java application to send messages to another application. It is highly scalable and allows us to loosely couple applications using asynchronous messaging. Using JMS we can read, send, and read messages.

**Benefits of using JMS with Spring Integration**

* **Load balancing:**Multiple consumers in separate virtual machine processes pull messages from a shared destination at a rate determined by their capabilities.
* **Scalability:**Adding enough consumer processes to avoid a backlog increases throughput and decreases response time.
* **Availability:**With multiple consumer processes, the overall system can remain operational even if one or more individual processes fail. Likewise, consumer processes can be redeployed one at a time to support a rolling upgrade.

**Here are some implementations of JMS is as follows:**

* Amazon SQS
* Apache ActiveMQ
* JBoss Messaging
* RabbitMQ

**JMS Message**

A JMS message can be divided into three parts that are as follows:

* **Header:**It contains the metadata about the message.
* **Properties:**It can further be subdivided into three sections –
  + **Application:**The java application sending message.
  + **Provider:**It is used by the JMS provider and is implementation-specific.
  + **Standard Properties:**These are defined by the JMS API.
* **Payload:**This field is the message itself.

