Active MQ:

* What is Message-oriented middleware
* Active Mq
* Advantages vs disadvantages

### Message queuing - a simple use case

Imagine that you have a web service that receives many requests every second, where no request can get lost, and all requests need to be processed by a function that has a high throughput. In other words, the web service always has to be highly available and ready to receive a new request instead of being locked by the processing of previously received requests.

In this case, placing a queue between the web service and the processing service is ideal. The web service can put the "start processing" message on a queue and the other process can take and handle messages in order. The two processes are decoupled from each other and do not need to wait. If you have a lot of requests coming in a short amount of time, the processing system will be able to process them all. The queue will persist with the requests even if their number grows.

Then imagine that the business and workload are growing and the system needs to be scaled up. All that needs to be done is to add more consumers to work off the queues faster.

How does ActiveMQ work?

Written in Java, [ActiveMQ](http://activemq.apache.org/) translates messages from sender to receiver. It can connect multiple clients and servers and allows messages to be held in queue, instead of requiring both the client and server to be available simultaneously in order to communicate. Messaging can still happen even if one application is temporarily indisposed.