**JMS(Java Message Service)：面向消息的中间件**

**两种模型：点对点(PTP，Point to Point)，发布/订阅(Pub/Sub)。**

**主要对象：**

**ConnectionFactory：用于创建客户端和消息提供商之间的连接。**

**Destination：消息发送的目的地，也是接收消息的来源。**

|  |  |  |
| --- | --- | --- |
| **公共接口** | **PTP专用** | **Pub/Sub专用** |
| **ConnectionFactory** | **QueueConnectionFactory** | **TopicConnectionFactory** |
| **Connection** | **QueueConnection** | **TopicConnection** |
| **Destination** | **Queue** | **Topic** |
| **Session** | **QueueSession** | **TopicSession** |
| **MessageProducer** | **QueueSender** | **TopicPublisher** |
| **MessageConsumer** | **QueueReceiver**  **QueueBrowser** | **TopicConsumer** |

**流程：**

**ConnectionFactory** **----> Connection ----> Seesion ----> MessageProducer、MessageConsumer、Message，MessageProducer、MessageConsumer通过Destination发送和接收Message。**

**在这几个对象中，ConnectionFactory、Connection、Destination可以用于多线程；Session、MessageProducer、MessageConsumer必须单线程。这是因为：一、Session是支持事务的JMS实体，实现多线程的事务是非常困难的。二、Session支持异步消息。**

**异步接收消息：MessageListener监听。**

**Broker：服务器端，可独立运行，也可内嵌。**

**JMX(Java Management Extension，Java管理扩展)**

**JMX在Java编程语言中定义了应用程序以及网络管理和监控的体系结构、设计模式、应用程序接口以及服务。通常使用JMX来监控系统的运行状态或管理系统的某些方面，比如清空缓存、重新加载配置文件等。**

**JMX体系结构分为以下四个层次：**

**设备层(**Instrumentation Level**)：主要定义了信息模型。在JMX中，各种管理对象以管理构件的形式存在，需要管理时，向MBean管理器进行注册。该层还定义了通知机制和一些辅助元数据等。**

**代理层(Agent Level)：主要定义了各种服务以及通信模型。该层的核心是一个MBean服务器。注册在MBean中的管理构件并不直接和远程应用程序通信，他们通过协议适配器和连接器进行通信。而这两者也以管理构件的形式存在。**

**分布服务层(**Distributed Service Level**)：主要定义了能对代理层进行操作的管理接口和构件，这样管理者就可以操作代理。JMX规范并没有给出该层的具体规范。**

**接口和类：**

**MBeanServerFactory、MBeanServer：MBean的服务器和工程类。**

**MBean：管理构件。各种Adapter也是MBean。**

**Notification：消息包，相当于JMS中的Message；Notification BroadCaster：广播器，将Notification发送出去；Notification Listener：Notification的监听器，用于接收消息。**

**几种MBean：**

**标准MBean：以MBean结尾的接口。**

**动态MBean：利用构造函数类MBeanConstructorInfo、属性类MBeanAttributeInfo、方法类MbeanOperationInfo来完成功能。自身需实现DynamicMBean接口。**

**模型MBean：构造出一个RequiredModelMBean。**