Topics: Java Mail & JMS Service

**Java Mail**

The **JavaMail** is an API that is used to compose, write and read electronic messages (emails).

The JavaMail API provides protocol-independent and plateform-independent framework for sending and receiving mails.

The **javax.mail** and **javax.mail.activation** packages contains the core classes of JavaMail API.

The JavaMail facility can be applied to many events. It can be used at the time of registering the user (sending notification such as thanks for your interest to my site), forgot password (sending password to the users email id), sending notifications for important updates etc. So there can be various usage of java mail api.

**Protocols used in JavaMail API**

|  |
| --- |
| There are some protocols that are used in JavaMail API.   * SMTP(Simple mail transfer protocol) * POP(Post office Protocol) * IMAP(Internet message Access protocol) * MIME(Multiple Internet mail extension) * NNTP() and others |

**SMTP**

SMTP is an acronym for Simple Mail Transfer Protocol. It provides a mechanism to deliver the email. We can use Apache James server, Postcast server, cmail server etc. as an SMTP server. But if we purchase the host space, an SMTP server is bydefault provided by the host provider. For example, my smtp server is mail.javatpoint.com. If we use the SMTP server provided by the host provider, authentication is required for sending and receiving emails.

**POP**

POP is an acronym for Post Office Protocol, also known as POP3. It provides a mechanism to receive the email. It provides support for single mail box for each user. We can use Apache James server, cmail server etc. as an POP server. But if we purchase the host space, an POP server is bydefault provided by the host provider. For example, the pop server provided by the host provider for my site is mail.javatpoint.com. This protocol is defined in RFC 1939.

**IMAP**

IMAP is an acronym for Internet Message Access Protocol. IMAP is an advanced protocol for receiving messages. It provides support for multiple mail box for each user ,in addition to, mailbox can be shared by multiple users. It is defined in RFC 2060.

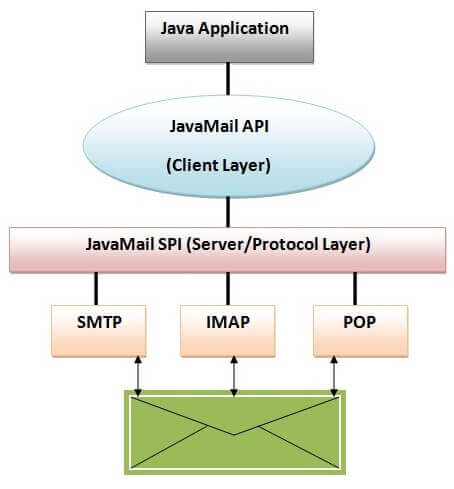
**MIME**

|  |
| --- |
| Multiple Internet Mail Extension (MIME) tells the browser what is being sent e.g. attachment, format of the messages etc. It is not known as mail transfer protocol but it is used by your mail program. |

**NNTP and Others**

There are many protocols that are provided by third-party providers. Some of them are Network News Transfer Protocol (NNTP), Secure Multipurpose Internet Mail Extensions (S/MIME) etc.

Mail Architecture



<https://static.javatpoint.com/images/Javamail1.jpg>

**JavaMail API Core Classes**

There are two packages that are used in Java Mail API: javax.mail and javax.mail.internet package. These packages contains many classes for Java Mail API. They are:

* javax.mail.Session class
* javax.mail.Message class
* javax.mail.internet.MimeMessage class
* javax.mail.Address class
* javax.mail.internet.InternetAddress class
* javax.mail.Authenticator class
* javax.mail.PasswordAuthentication class
* javax.mail.Transport class
* javax.mail.Store class
* javax.mail.Folder class etc.

Qusestion:

* How to send and receive email using JavaMail API ?
* How to send email through gmail server ?
* How to send and receive email with attachment ?
* How to send email with html content including images?
* How to forward and delete the email ?

JMS Service

JMS (Java Message Service) is an API that provides the facility to create, send and read messages. It provides loosely coupled, reliable and asynchronous communication.

JMS is also known as a messaging service.

**Understanding Messaging**

Messaging is a technique to communicate applications or software components.

JMS is mainly used to send and receive message from one application to another.

**Requirement of JMS**

Generally, user sends message to application. But, if we want to send message from one application to another, we need to use JMS API.

Consider a scenario, one application A is running in INDIA and another application B is running in USA. To send message from A application to B, we need to use JMS.

**Advantage of JMS**

1) **Asynchronous:** To receive the message, client is not required to send request. Message will arrive automatically to the client.

2) **Reliable:** It provides assurance that message is delivered.

**Messaging Domains**

There are two types of messaging domains in JMS.

Point-to-Point Messaging Domain

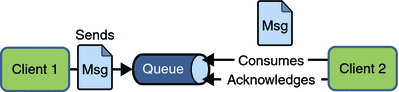
Publisher/Subscriber Messaging Domain

**1) Point-to-Point (PTP) Messaging Domain**

In PTP model, one message is **delivered to one receiver** only. Here, **Queue** is used as a message oriented middleware (MOM).

The Queue is responsible to hold the message until receiver is ready.

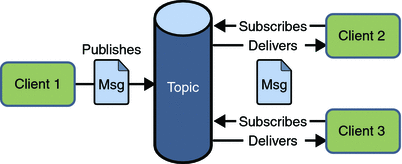
In PTP model, there is **no timing dependency** between sender and receiver.



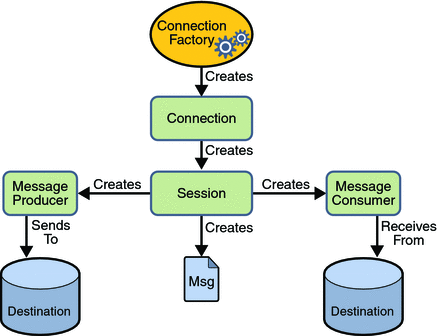
**2) Publisher/Subscriber (Pub/Sub) Messaging Domain**

In Pub/Sub model, one message is **delivered to all the subscribers**. It is like broadcasting. Here, **Topic** is used as a message oriented middleware that is responsible to hold and deliver messages.

In PTP model, there is **timing dependency** between publisher and subscriber.



**JMS Programming Model**



JMS Queue Example

To develop JMS queue example, you need to install any application server. Here, we are using glassfish3 server where we are creating two JNDI.

Create connection factory named myQueueConnectionFactory

Create destination resource named myQueue

**Message Driven Bean**

A message driven bean (MDB) is a bean that contains business logic. But, it is invoked by passing the message. So, it is like JMS Receiver.

MDB asynchronously receives the message and processes it.

A message driven bean receives message from queue or topic, so you must have the knowledge of JMS API.

A message driven bean is like stateless session bean that encapsulates the business logic and doesn't maintain state.

**Entity Bean in EJB 3.x**

Entity bean represents the persistent data stored in the database. It is a server-side component.

In EJB 2.x, there was two types of entity beans: bean managed persistence (BMP) and container managed persistence (CMP).

Since EJB 3.x, it is deprecated and replaced by JPA (Java Persistence API) that is covered in the hibernate tutorial.