JavaMail API Core Classes

The `javax.mail` and `javax.mail.internet` package consist the commonly used core classes for JavaMail API.

Commonly used core classes of JavaMail API:

1. `javax.mail.Session`
2. `javax.mail.Message`
3. `javax.mail.internet.MimeMessage`
4. `javax.mail.Address`
5. `javax.mail.internet.InternetAddress`
6. `javax.mail.Authenticator`
7. `javax.mail.Transport`
8. `javax.mail.Store`
9. `javax.mail.Folder`

1. `javax.mail.Session`:

The `javax.mail.Session` class is the primary class for JavaMail API. The `Session` object is used to handle configuration setting and authentication and it acts as the connection factory for the JavaMail API.

We can get the `Session` object from `getDefaultInstance()` or `getInstance()` method of `Session` class.

`getDefaultInstance()`: This method returns the default session.

**Syntax of `getDefaultInstance()`:**

```
public static Session getDefaultInstance()
```

```
public static Session getDefaultInstance(Properties props)
```

```
public static Session getDefaultInstance(Properties props, Authenticator auth)
```

`getInstance()`: This method returns the new session.

**Syntax of `getInstance()`:**

```
public static Session getInstance()
```

```
public static Session getInstance(Properties props)
```

```
public static Session getInstance(Properties props, Authenticator auth)
```
public static Session getInstance(Properties props, Authenticator auth)

2. javax.mail.Message:

The javax.mail.Message class is used to create or compose an email message. It is an abstract class so its subclass javax.mail.internet.MimeMessage is mostly used.

To create a message we have to pass session object in MimeMessage class constructor.

MimeMessage message=new MimeMessage(session);

After creating the message object we can store information in it. MimeMessage class provides the methods to store the information in it.

Commonly used methods of MimeMessage class:

1. setFrom(Address address): It is used to store the from header field.
   
   Syntax: public void setFrom(Address address)

2. addRecipients(Message.RecipientType type, String addresses): It is used to add the given address to the recipient type.
   
   Syntax: public void addRecipient(Message.RecipientType type, Address address)

3. addRecipients(Message.RecipientType type, Address[] addresses): It is used to add the given address to the recipient type.
   
   Syntax: public void addRecipients(Message.RecipientType type, Address[] addresses)

4. setSubject(String subject): It is used to set the subject header field.
   
   Syntax: public void setSubject(String subject)

5. setText(String textmessage): It is used to set the text as the message content using text/plain MIME type.
   
   Syntax: public void setText(String textmessage)

6. setContent(Object msg, String contentType): It is used to set the text as the message content using text/plain MIME type.
   
   Syntax: public void setContent(Object msg, String contentType)

Three predefined address types in Message class:

1. Message.RecipientType.TO
2. Message.RecipientType.CC
3. Message.RecipientType.BCC

3. javax.mail.internet.MimeMessage:

The javax.mail.internet.MimeMessage class is the subclass of javax.mail.Message class. A MimeMessage is an e-mail message that understands MIME types and headers.
4. `javax.mail.Address`:

The `javax.mail.Address` class is an abstract class that defines the addresses like from, to etc in a message. As it is an abstract class so its subclass `javax.mail.internet.InternetAddress` is mostly used.

We can create an Address object by passing email address or name along with the email address.

**Syntax:**

```
Address address = new InternetAddress("emailAddress");
Address address = new InternetAddress("emailAddress", name);
```

5. `javax.mail.internet.InternetAddress`:

The `javax.mail.internet.InternetAddress` is the subclass of `javax.mail.Address` class. It represents an Internet email address using the syntax of RFC822.

6. `javax.mail.Authenticator`:

The `javax.mail.Authenticator` is an abstract class which is used to protect mail resources on the mail server. `PasswordAuthentication` is its subclass and we can create a `PasswordAuthentication` object by passing username and password in its constructor.

**Syntax:**

```
PasswordAuthentication auth = new PasswordAuthentication("userName", "password")
```

7. `javax.mail.Transport`:

The `javax.mail.Transport` is an abstract class which is used to send the message.

**Syntax:**

```
Transport.send(message);
```

8. `javax.mail.Store`:

The `javax.mail.Store` is an abstract class that models a message store and its access protocol, for storing and retrieving messages.

9. `javax.mail.Folder`:

The `javax.mail.Folder` is an abstract class that represents a folder for mail messages. A folder can contain subfolders.

Next Topic: JavaMail API sending email. (http://tutorialspointexamples.com/javamail-api-sending-email/)

Previous Topic: JavaMail API overview. (http://tutorialspointexamples.com/javamail-api-overview/)

**Related Topics:**

- How to build java project using ant in eclipse? (http://tutorialspointexamples.com/how-to-build-java-project-using-ant-in-eclipse/)
JAXB marshalling – convert java object to xml example. (http://tutorialspointexamples.com/jaxb-marshalling-convert-java-object-to-xml-example-using-one-pojo/)


Generics class example. (http://tutorialspointexamples.com/generics-class-example/)

OGNL in struts 2. (http://tutorialspointexamples.com/ognl-in-struts-2/)

Hibernate One-to-One Mapping using xml. (http://tutorialspointexamples.com/hibernate-one-to-one-mapping-using-xml/)

Send inline image in email using JavaMail API. (http://tutorialspointexamples.com/send-inline-image-in-email-using-javamail-api/)

Quartz 2 JobListener example. (http://tutorialspointexamples.com/quartz-2-joblistener-example/)

JavaMail API Tutorial

JavaMail API overview. (http://tutorialspointexamples.com/javamail-api-overview/)
JavaMail API core classes. (http://tutorialspointexamples.com/javamail-api-core-classes/)
JavaMail API sending email. (http://tutorialspointexamples.com/javamail-api-sending-email/)
Send email. (http://tutorialspointexamples.com/send-simple-email-using-javamail-api/)
Send email with attachment. (http://tutorialspointexamples.com/send-email-with-attachment-using-javamail-api/)
Send HTML content. (http://tutorialspointexamples.com/send-html-content-in-email-using-javamail-api/)
Send inline image. (http://tutorialspointexamples.com/send-inline-image-in-email-using-javamail-api/)
Send email through Gmail server using TLSSconnection. (http://tutorialspointexamples.com/send-simple-email-through-gmail-server-using-tls-connection/)
Send email through Gmail server using SSL connection. (http://tutorialspointexamples.com/send-simple-email-through-gmail-server-using-ssl-connection/)
Receiving emails. (http://tutorialspointexamples.com/receiving-or-fetching-simple-email-using-javamail-api/)
Receiving emails with attachments. (http://tutorialspointexamples.com/receiving-or-fetching-email-with-attachment-using-javamail-api/)
Replying emails. (http://tutorialspointexamples.com/replying-emails-using-javamail-api/)
Forwarding emails. (http://tutorialspointexamples.com/forwarding-emails-using-javamail-api/)
Deleting emails. (http://tutorialspointexamples.com/deleting-emails-using-javamail-api/)