Receiving Emails

The common process for receiving emails has the following steps:

1. Retrieve a set of messages from the server.
2. Process the returned set of messages.
3. Delete all or some messages, or move them to a specific folder.

Operations expecting a **folderPath** argument will treat this the following way:

Folder Path	Description	Example
NULL	This is interpreted as meaning the `<Inbox>` folder.	
path beginning with /	This is interpreted as a path from the mail connections root folder.	`/Archive/Orders`
path not beginning with /	This is interpreted as a relative path to the `<Inbox>` folder, so translates to `/<Inbox>/Archive/Orders`.	`Archive/Orders`

The intention of this behavior is to abstract away the fact that `<Inbox>` has many different names depending on the connection, i.e. IMAP typically uses `INBOX`, Exchange calls it `Inbox`, and a German Exchange calls it `Posteingang`.

Also note that **POP3 does not support folders** other than `<Inbox>`. Hence when you specify a POP3 connection, **folderPath** parameters other than NULL will raise an exception.

On this Page:

Related Pages:

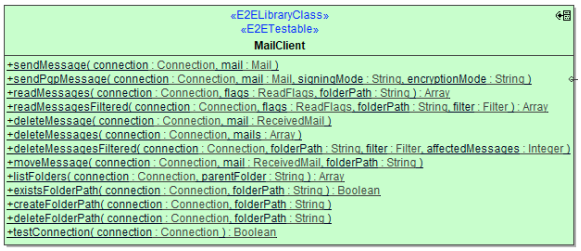
- Mail Server Connection
- JavaMail Library Reference

Step 1: Retrieving Messages

You can retrieve messages using one of

- readMessages
- readMessagesFiltered

Both return an **Array** of **ReceivedMail** (see Processing Messages further below).



Operations

readMessages

Optional: IPgpKeyProvider Implementation

For PGP decryption to work, be sure to register your implementation of IPgpKeyProvider (see Keys and Certificates). If not provided, PGP encrypted messages will still be retrieved, with two attachments that contain the encrypted message information.

Parameter	Type	Direction	Description

connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).
flags	ReadFlags	in	Specify an instance of **ReadFlags** to set some mail reading flags.
folderPath	String	in	Specify a path to a folder to read from. See note regarding folder handling.

readMessagesFiltered

> **Optional: IPgpKeyProvider Implementation**
>
> For PGP decryption to work, be sure to register your implementation of IPgpKeyProvider (see Keys and Certificates). If not provided, PGP encrypted messages will still be retrieved, with two attachments that contain the encrypted message information.

Parameter	Types	Direction	Description
connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).
flags	ReadFlags	in	Specify an instance of **ReadFlags** (see below).
folderPath	String	in	Specify a path to a folder to read from. See note regarding folder handling.
filter	Filter	in	Specify an instance of **Filter** containing the filter criteria.

Types

ReadFlags

Attribute Name	Type	Description	Allowed Values	Comment
maxResultCount	Integer	Specify a limit for the number of messages to be returned. If you do not provide a filter as parameter, it makes sense to provide a meaningful limit, as the complete result set will be kept in memory.	any positive integer	Default is NULL, meaning no limit is imposed.
deleteOnServer	Boolean	Specify whether successfully read messages will automatically be deleted on the server.	true	Delete messages on server after successful read. Messages that cannot be read successfully will not be deleted.
			false	Do not delete messages on server (default).
markAsRead	Boolean	Specify whether successfully read messages will automatically be marked as read on the server.	true	Mark messages as read on server after successful read. Messages that cannot be read successfully will not be marked read.
			false	Do not mark messages read on server (default).
withContent	Boolean	Specify whether the message body (HTML and/or plain text) should be populated.	true	Populate message body.
			false	Message body is empty (default).
withAttachments	Boolean	Specify whether attachments should be retrieved.	true	Retrieve attachments.
			false	Do not retrieve attachments (default).

withEmbeddedMails	Boolean	Specify whether attachments with mime type message/rfc822 should be retrieved as embedded messages. Requires withAttachments to be true to have an effect.	true	Retrieve attachments with mime type message /rfc822 as embedded messages.
			false	Do not retrieve attachments with mime type message/rfc822 as embedded messages, but as regular attachments (default).
withHeaders	Boolean	Specify whether messaging headers should be populated.	true	Populate messaging headers.
			false	Messaging headers are empty (default).
withRawContents	Boolean	Specify whether the raw message source should be retrieved.	true	Retrieve raw message source. This is similar to "view source" on most email clients, hence the output can be huge.
			false	Do not retrieve raw message source (default).
verifySignatures	Boolean	Specify whether for any signed messages a verification of the signature should be attempted.	true	Verify signature. To successfullly verify PGP signatures, a public key ring must be provided with the connection parameter.
			false	Do not verify signature (default).

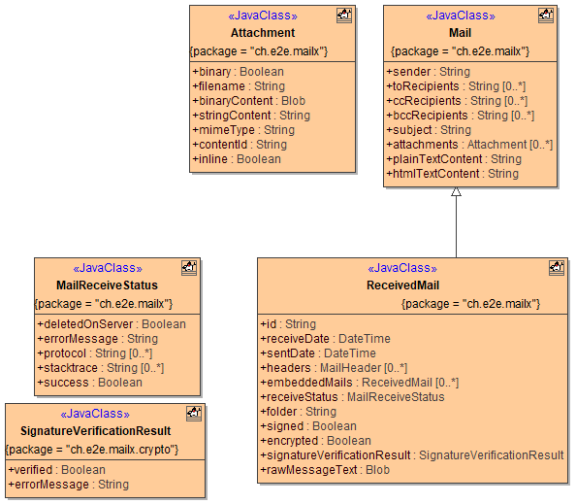
Filter

Attribute Name	Type	Description	Allowed Values	
unreadOnly	Boolean	Specify whether only unread messages should be returned.	true	Return unread messages only.
			false	Return all messages (default).
fromTimestamp	DateTime	Specify the beginning of a date range that should be used to retrieve messages (see also **toTimestamp** below). Make sure that **fromTimestamp** and **toTimestamp** do not contradict each other as this is not checked.	any DateTime	
toTimestamp	DateTime	Specify the end of a date range that should be used to retrieve messages (see also **fromTimestamp** below). Make sure that **fromTimestamp** and **toTimestamp** do not contradict each other as this is not checked.	any DateTime	
subjectSubstring	String	Specify a filter substring for the email subject. Only messages containing this string as a literal substring are returned. **subjectSubstring** is case sensitive.	any String with printable characters (case sensitive)	

sender	String	Specify a dedicated sender you want to retrieve messages from. Only message sent from this address are returned	any valid email address (case insensitive)
senderRegex	String	Specify a regex filter to be applied to the mail sender. Only messages with the sender matching the filter expression are returned.	any valid regular expression (Java style)
subjectRegex	String	Specify a regex filter to be applied to the mail subject. Only messages with the subject matching the filter expression are returned.	any valid regular expression (Java style)
attachmentNameRegex	String	Specify a regex filter to be applied to the mail attachments. Only messages with at least one attachment's name matching the filter expression are returned. Make sure that you are using **withAttachments** = true when reading the mails (see ReadFlags).	any valid regular expression (Java style)

Step 2: Processing Messages

You can process emails as you need. Retrieved messages are delivered as an **Array** of **ReceivedMail** objects, which inherit from **Mail** and add some more attributes available after receiving:



Classes

ReceivedMail

Attribute Name	Type	Description	Possible Values
sender	String	Contains the email address of the sender.	
toRecipients	Array of String	Contains an array of TO recipient's email addresses.	
ccRecipients	Array of String	Contains an array of CC recipient's email addresses.	
bccRecipients	Array of String	Contains an array of BCC recipient's email addresses.	
subject	String	Contains the email subject.	
attachments	Array of Attachment	Contains an array of email attachments. If this is not populated, you may have set **withAttachments** = false when reading the mails (see ReadFlags).	

plainTextContent	String	Contains the plain text message if available. Mime messages may contain HTML, Plain Text, or both.		
htmlTextContent	String	Contains the HTML message if available. Mime messages may contain HTML, Plain Text, or both.		
id	String	Contains the unique ID of the message within a folder on the mail server.		
receiveDate	DateTime	Contains a timstamp indicating when the message arrived on the server.		
sentDate	DateTime	Contains a timstamp indicating when the message has been sent.		
headers	Array of Mail Header	Contains an array of internet email headers. If this is not populated, you may have set **withHeaders** = false when reading the mails (see ReadFlags).		
embeddedMails	Array of ReceivedMail	Contains an array of nested/embedded mail objects (attached email messages, e.g. .eml or .msg attachments). If this is not populated, you may have set **withAttachments** = false when reading the mails (see ReadFlags).		
receiveStatus	MailReceiveStatus	Contains detailed information about the receive status of the email.		
folder	String	Contains the folder the message was read from.	NULL	`<Inbox>`.
			folder path	The folder path below `<Inbox>`.
signed	Boolean	Contains the signing status of the message.	true	Message is signed.
			false	Message is not signed.
encrypted	Boolean	Contains the encryption status of the message. detection is supported, however receiveStatus.success will always be false, as decryption is not yet supported	true	Message is encrypted.
			false	Message is not encrypted.
signatureVerificationResult	SignatureVerificationResult	Contains detailed information about the verification of signatures.		
rawMessageText	Blob	Contains the raw message. If this is not populated, you may have set **withRawContents** = false when reading the mails (see ReadFlags).		

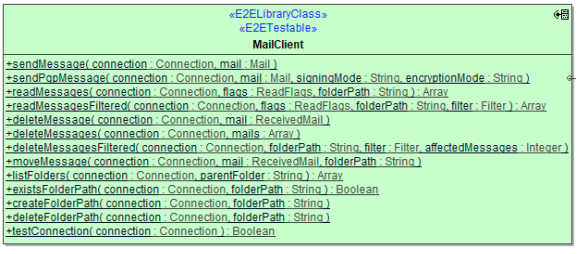
Attachment

Attribute Name	Type	Description	Possible Values	
binary	Boolean	Indicates whether **binaryContent** or **stringContent** is used.	true	Attachment contents canl be found in, or shall be taken from, **binaryContent**.

			false	Attachment contents can/shall be found in, or shall be taken from, **stringContent**.
filename	String	The filename of the attachment.	• can be NULL **Sending Messages** This is the name of the file as you want it to appear at the recipient's side, or vice versa. Do not provide a local file name here and expect the library to load the attachment from that file. Use the File System Adapter instead, and provide the attachment as binary blob content (see **binaryContent**).	
binaryContent	Blob	Contains the binary content.	any Blob Either one of **binaryContent** or **stringContent** is provided, depending on **binary**.	
stringContent	String	Contains the non-binary content.	any String Either one of **binaryContent** or **stringContent** is provided, depending on **binary**.	
mimeType	String	Contains the mime type (content type) of the attachment.	any valid MIME content-type, e.g. `text/plain`, `image/png`, `application/octet-stream` etc. **mimeType** is unreliable upon receiving emails, i.e. can be NULL. Especially when reading from Exchange servers using **exchange** or **office365** connections.	
contentId	String	The content id of the attachment, a unique id across all attachments of the same message.	When set to e.g. `myUniqueId`, you can refer to this attachment from HTML message content like `<img src="cid:myUniqueId">`	
inline	Boolean	Controls whether the attachment is flagged with the corresponding content disposition tag.	true	Attachment is flagged, and will be displayed inline within a HTML message.
			false	Attachment is not flagged.

Step 3: Deleting/Moving Messages

After messages have been retrieved, they are often deleted or moved to a specific folder.



Deleting Messages

Deleting messages is done by any one of the following operations:

- deleteMessage
- deleteMessages
- deleteMessagesFiltered

deleteMessage and **deleteMessages** operate on an instance of **ReceivedMail**, means you can call them on objects returned by a read on the mailbox. **deleteMessagesFiltered** operates on a folder and a filter criterion. It deletes all matching messages.

> Deletion is always "hard" deletion. Mail items are not moved to "Trash" or "Deleted Items". Once deleted, the message is gone.

deleteMessage

Parameter	Types	Direction	Description
connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).
mail	ReceivedMail	in	Provide a previously received mail object.

deleteMessages

Parameter	Types	Direction	Description
connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).
mail	Array of ReceivedMail	in	Provide an array previously received mail object.

deleteMessagesFiltered

Parameter	Types	Direction	Description	Allowed Values / Example
connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).	
folderPath	String	in	Specify a path to a folder to delete from. See note regarding folder handling.	`orders/incoming`

filter	Filter	in	Specify an instance of **Filter** containing the filter criteria. ⚠ If no filter criteria are set, i.e. if you provide an empty filter, all messages will be deleted.	
affectedMessages	Integer	out	Returns the number of deleted messages.	This might be lower than the total number of messages matching the filter if that number is huge, as mail servers may restrict the maximum number of messages processed in one call. If you want to be sure that all messages matching the filter are actually deleted, call this operation in a loop until `affectedMessages` becomes 0.

Moving Messages

Moving messages operates in instances of **ReceivedMail**, objects returned by a read on the mailbox. You can specify the target folder to move the message to.

Parameter	Types	Direction	Description	Allowed Values / Example
connection	Connection	in	Specify the **Connection** object defining the mail server connection parameters to use (see Mail Server Connection).	
mail	ReceivedMail	in	Provide an array previously received mail object.	
folderPath	String	in	Specify a path to a specific sub-folder to move the message to. See note regarding folder handling. POP3 connections will cause an exception to be thrown here, as POP3 does not support the concept of folders.	`orders/incoming/accepted`