Documentation for the

FB-MailService

FB-MailService

This document describes the FB-MailService, WebServiceClient and ProcessStarter software.

In order to use this software always follow the instructions, notes and information in this documentation.
This document and the software are protected by copyright. Both the document and the software are subject of the License Agreement and must only be used in accordance with the provisions stated there.

State of the document: December 22, 2021

FB Software – Dipl.-Inf. Felix Billau

Contents

1   Overview

The FB-MailService is a mail server with groupware functionality. All components (mail delivery agent, mail transfer agent, spam filter, POP3 client, HTTP, POP3, SMTP, mail and all other servers) are integrated in one application. The implementation language used is Java.

1.1   Interfaces for clients

For two types of mail clients, the FB-MailService has interfaces: for Web Service clients and for POP3 clients.

A Web Service based mail client (such as the included WebServiceClient) works completely "on the server".
This has the advantage of being able to access mails from anywhere with any client on any device and always get the same view of the mails.

POP3 clients, however, retrieve the emails from the FB-MailService and store them locally.

Mail sending also takes place using the FB-MailService. The FB-MailService acts as an SMTP server and receives the mail from authenticated mail clients. Depending on the configuration, the mail is then sent using a specific SMTP server or the SMTP server of the FB-MailService contacts the recipient SMTP server directly. Sending the mails using the FB-MailService makes sense to be able to add the recipient addresses of sent mails to the whitelist of the corresponding account.

1.2   Inbound mail server

As an inbound mail server, the SMTP server of the FB-MailService is responsible for one or more domains itself.
Submitting SMTP servers (MTAs) contact the SMTP server of the FB-MailService and can send them the mail for **local** delivery, but not for forwarding (non-local delivery).
The FB-MailService is basically designed so that it can not be used as an open relay. As an inbound mail server, it accepts mails only if

- it is responsible for the global part of the recipient email address (domain - the part of the mail address after the @ sign) to which the mail is addressed.

- it knows the local part of the recipient email address (the part before the @ sign), ie. there is a mailbox or alias of that name.

An acceptance of mails, for whose recipient domain the FB-MailService is not responsible, only takes place, if submitting mail clients have previously authenticated (see 1.3 - Mail delivery).

1.3   Mail delivery

Sending mail clients contact the SMTP server of the FB-MailService. They authenticate themselves and hand over the mail to be delivered.

Depending on the setting, the SMTP server of the FB-MailService then contacts the recipient SMTP server directly, or it forwards the mails to a specific SMTP server, which then has to take care of further delivery.
This can be configured differently for each account, ie. or each account, you can set individually whether the mails are to be delivered directly or whether they should be forwarded to a specific SMTP server. In the latter case, it is also possible to specify a different SMTP server (and the possibly necessary authentication data) for each account.

The recipient addresses of the mail to be delivered are added to the respective accounts whitelist. This is useful to ensure that emails from recipients of a self-sent mail are not mistakenly classified as spam.

The SMTP server of the FB-MailService is basically designed in such a way that mail can not be sent to a PO Box for which the FB-MailService is not responsible without a previous authentication.
The SMTP server of the FB-MailService accepts mail for **non-local** delivery only if the client has previously authenticated.

In order to the FB-MailService can not be misused as spam sling, the number of **different** recipient addresses per day and account is adjustable. Default is 144.
This means that by default e-mails can only be sent to 144 **different** recipient addresses per day and account via the FB-MailService.

For clarification: If this setting is set to 144, a maximum of 144 different recipients per day and account can be contacted. But as often as you like. This setting does **not** limit the number of deliverable mails per day and account to 144, but only the number of different recipients that can be written to per day and account.
If this limit is too low, the administrator must explicitly think about how many **different** recipient addresses he wants to allow per day and account.

1.4   POP3 retrieval

Towards a retrieving POP3 client, the FB-MailService acts as POP3 server. The exact procedure for receiving mail is as follows:

- A retrieving POP3 client contacts the POP3 server of the FB-MailService and authenticates itself.

- The FB-MailService retrieves the mails from all configured POP3 servers of the FB-MailService account and **deletes** them there.

- The retrieved mails are saved on the local hard disk, either in the `Inbox` or in the `Spam` folder of the respective account, ie. the hard disk of the computer on which the FB-MailService is running.

- All emails in `Inbox` are then made available to the retrieving mail client using the POP3 protocol.

- The FB-MailService can be configured to additionally periodically check the servers of the configured POP3 accounts in a freely adjustable interval of 1 to 1440 minutes for existing mails, retrieve them if necessary and filter them immediately. This can lead to a noticeably faster retrieval process for querying mail clients, since the mail filtering has already been carried out.

2   System Requirements

Supported Operating Systems The FB-MailService can be used on all platforms for which a current Java Runtime Environment (or Java Development Kit) is available. Among others, these are the following operating systems:

- Windows
- Linux
- Solaris
- Mac OS X
- (Free, Net and Open) BSD
- AIX

Required hardware The required amount of memory, processors, type and amount of disk space and bandwidth of the network connection depends on the number of MailService accounts, how frequented these accounts are and the desired response time. Due to its efficiency, the equipment of an average home computer should always be sufficient in the vast majority of deployment scenarios for the FB-MailService.

Required Software A current Java Runtime Environment (with all available security updates) must be installed.

Compatibility In principle, all POP3 and SMTP compliant mail servers and mail clients as well as all web service standard compliant clients should be compatible with the FB-MailService.

Required Ports Free ports are required for the following services:

- For the POP3 server of the FB-MailService (default: Port 110 - can be changed)
- If offered for the "POP3 over SSL" server (POP3S server, default: port 995 - can be changed)
- For the SMTP server of the FB-MailService (Default: Port 25 - can be changed)
- If offered for the "SMTP over SSL" server (SMTPS server, default: port 465 - can be changed)
- For the HTTP server of the FB-MailService (Default: Port 80 - can be changed)
- If offered for the "HTTP over SSL" server (HTTPS server, default: port 443 - can be changed)

3   Install/Uninstall

3.1   Install

Note: Depending on where the FB-MailService is to be installed, you may need administrator or root privileges to install it.
The installation is done by a graphical user interface. If no graphics environment is available on the target system, the installation can also be performed on another computer with graphics environment and the installation folder then be moved to the target system.

The installation is started by starting the executable installation jar file `FB-MailService-Install.jar`. The installation takes place in four steps:

1. Selection of the language for the installation.

2. Approval of the license agreement. To install and run the FB-MailService, must be approved to the license agreement.

3. Selection of the installation folder and the first mail store. Here are two options possible:

   (a) The folder of the first mail store is the installation folder.
   (b) The folder of the first mail store is a separate folder.

   The folders of the accounts are created in the mail stores. In it, the account configuration, the address book, the calendar, the uploaded files, blacklist and whitelist and all mails of an account are saved.

   Note: The paths of the here specified and possibly all further mail stores can be changed at any time in the configuration file.

4. Perform the installation.

Note: The FB-MailService must not be **installed** over an already existing installation of a FB-MailService having a different hash value, since no compatibility between the versions is guaranteed. Something may have changed about the encryption algorithm or the format of the `fbms` files. This means that compatibility must always have been extensively tested on a recent backup.

3.2   Uninstall

To uninstall the FB-MailService, stop the FB-MailService and simply delete the installation folder. During the installation, no further changes are made to the system, so the deinstallation of the FB-MailService is complete by deleting the installation folder and the mail stores.

3.3   Start/Stop

The FB-MailService is started by starting the executable Jar file MailService.jar. On some operating systems (eg Windows) this can be done by double-clicking on this file.
The general procedure is as follows:

- In the console change to the root directory of the FB-MailService.

- Execute the FB-Mail service by typing `java -jar MailService.jar` and pressing "Enter". This works if the Java Bin directory has been added to the "PATH" environment variable. Otherwise, the call must be preceded by the path to the Java bin directory:

  `/path/to/Java/bin/directory/java -jar MailService.jar`

  `"C:\path to Java bin directory\java.exe" -jar MailService.jar`

The FB-MailService is terminated either via the console using the command `exit` or by the context menu: Right-click on the icon of the FB-MailService in the system tray and in the context menu click on "Exit".

4   Control of FB-MailService

When the FB-MailService starts, the letter icon with the magnifying glass will be displayed in the system tray (if available) and all servers will be booted up according to the configuration.
This icon can be used to control the FB-MailService. Right-clicking on the icon displays the context menu.
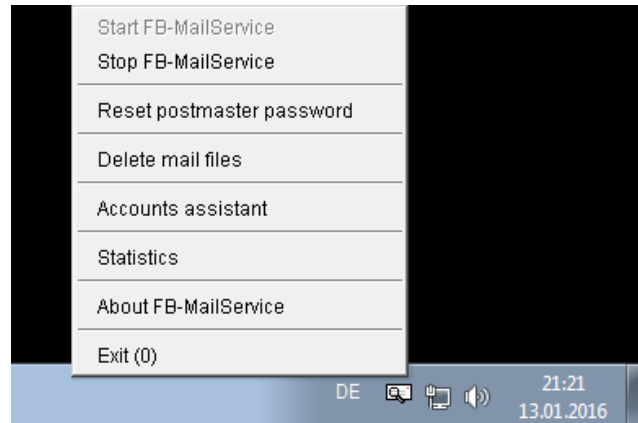
If no system tray is available, or if the FB-MailService is started using the -c option, in the console a command-line will be started instead, which can be used to control the FB-MailService.
The character coding for the outputs of the FB-MailService on the console is "Cp850" on Windows, otherwise "UTF-8".
The language of the output is the same as the system language.

The FB-MailService knows the following control commands (called by the context menu or the console):

Start FB-MailService If the servers and the mailsender of the FB-MailService have been shut down, they can be restarted by clicking on this menu item.
Corresponding command in the console: start

Stop FB-MailService By clicking on this menu item the servers and the mailsender of the FB-MailService can be shut down, so that no further querries can be sent to the FB-MailService and no new emails are sent. Active server and client threads are not aborted but will continue to be processed.
Corresponding command in the console: stop

Reset Postmaster Password Clicking on this menu item assigns a new password to the postmaster account and displays it. This is only possible locally by this menu item or the console, but not via the web service.
Corresponding command in the console: reset

Delete mail files By clicking this menu item, the actual deletion of files of deleted mails from the hard disk can be initiated.
Note: If "Delete mails at" is blank in the configuration file, the files of deleted mails are immediately removed from the hard disk and this command has no effect.
Corresponding command in the console: clean

Account Wizard Click on this menu item to start the Account Wizard, see 5.3 - The Account assistant.
Corresponding command in the console: accounts

Statistics By clicking on this menu item, the statistics gui is started where a statistical evaluation of FB-MailService statistics files can be made and graphically displayed, see 6 - Statistics.
Corresponding command in the console: stat

About FB-MailService Display license information.
Corresponding command in the console: about

Exit Clicking on this menu item ends the FB-MailService. All servers and mail senders are shut down, and any active server, client, and mail delivery threads are aborted. Next to "Exit" the number of threads is displayed in parentheses.
If you do not want to cancel any active threads before exiting the FB-MailService, you can first click on "Stop FB-MailService" and wait until the display next to "Exit" shows 0 active threads. However, the display can not be changed while the context menu is displayed. It must be called up from time to time to get an up-to-date display of the remaining active threads.
Corresponding command in the console: exit

In the console there is also the command **show**. It can display the number of all currently active server and client threads, broken down into HTTP, POP3 and SMTP server threads and POP3 client and mail delivery threads.

5   Configuration

The configuration of the FB-MailService is controlled by a configuration file. This file is located in the root directory of the FB-MailService and is called "Config.ini".
To change the configuration of the FB-MailService, this file must be edited.

The individual entries of an FB-MailService configuration file are described in the following section.

5.1   The configuration file

A configuration file for the FB-MailService consists of four sections:

- POP3 server

- SMTP server

- HTTP server

- Other settings

5.1.1   POP3 server

Offer POP3 Server Specifies whether the POP3 server should be offered. Valid values: `true` or `false`.

Local IP address to bind POP3 Server to Here you can specify the IP address of the network card to which the POP3 server is to be bound.
If left blank, the POP3 server binds to all existing network interfaces. Valid values: any valid IP address or empty.

POP3 port Specifies the POP3 port (default is 110). Valid values: 1 - 65535.

Offer POP3S Server Specifies whether the "POP3 over SSL" server (POP3S server) should be offered. Valid values: `true` or `false`.

Local IP address to bind POP3S server to Here you can specify the IP address of the network card to which the POP3S server is to be bound.
If left blank, the POP3S server binds to all existing network interfaces. Valid values: any valid IP address or empty.

POP3S port Specifies the POP3S port (default is 995). Valid values: 1 - 65535.

Path to keystore file To offer the POP3S server or the `STLS` command, the path to a keystore file[1] [2] that stores a certificate required for the TLS protocol needs to be specified.
If a relative path is specified, it is interpreted relative to the root directory of the FB-MailService. If the keystore file is in the root directory of the FB-MailService, only the name of the file needs to be specified here.
Note: Aliases for key pairs in the keystore must always have the same password as the keystore itself.

keystore format Specifies the format of the keystore. Valid values: `JCEKS`, `JKS` or `PKCS12`.

Keystore password Specification of the keystores password.

---

[1]http://download.oracle.com/javase/6/docs/technotes/tools/solaris/keytool.html
[2]http://download.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html

Refuse all clear-text login commands Since the POP3 server of the FB-MailService implements `STARTTLS` (by the POP3 command `STLS`), in accordance with RFC 2595[3] it must be be able to configure it to reject all plain text login commands.

If this option is set to `true`, all commands that result in plain text transmission of username or password will be rejected.

If the connection is not made via the POP3S server, or the `STLS` command fails (eg due to a missing keystore) then logins would only be available by `auth cram-md5` or `apop` methods, but not by `user` / `pass` method. Valid values: `true` or `false`.

5.1.2   SMTP server

Offer SMTP Server Specifies whether the SMTP server should be offered. Valid values: `true` or `false`.

Local IP address to bind SMTP Server to Here you can specify the IP address of the network card to which the SMTP server is to be bound.

If left blank, the SMTP server binds to all existing network interfaces. Valid values: any valid IP address or empty.

SMTP port Specifies the SMTP port (default is 25). Valid values: 1 - 65535.

Offer SMTPS server Specifies whether the "SMTP over SSL" server (SMTPS server) should be offered. Valid values: `true` or `false`.

Local IP address to bind SMTPS Server to Here you can specify the IP address of the network card to which the SMTPS server is to be bound.

If this setting is left blank, the SMTPS server is bound to all existing network interfaces. Valid values: any valid IP address or empty.

SMTPS port Specifies the SMTPS port (default is 465). Valid values: 1 - 65535.

Path to keystore file To provide the SMTPS server or the `STARTTLS` command, the path to a keystore file that stores a certificate required for the TLS protocol needs to be specified.

If a relative path is specified, it is interpreted relative to the root directory of the FB-MailService. If the keystore file is in the root directory of the FB-MailService, only the name of the file needs to be specified here.

keystore format Specifies the format of the keystore. Valid values: `JCEKS`, `JKS` or `PKCS12`.

Keystore password Specification of the keystores password.

Fully qualified host name The fully qualified name under which the SMTP server of the FB-MailService is reached. Default is `localhost`.

Note: If e-mails are to be sent directly from the FB-MailService, this information should be identical to the result of a reverse DNS request to the IP address of the computer on which the FB-MailService runs (ie being the same as the DNS-PTR-resource record for this IP address) as it will be used in `HELO` (or `EHLO`) when logging in to other SMTP servers. This reduces the spam probability of own mails.

TLS setting Specifies whether the `STARTTLS` command should be executed for directly sent mails if it is offered by the receiving SMTP server. 0 means the command is not executed, 1 means the command will be executed if offered. Default is 1.

Note: For mail sent using a specific SMTP account, the account settings apply.

---

[3]http://tools.ietf.org/html/rfc2595

Maximum mail size (in megabytes) Specify the maximum mail size in megabytes that the SMTP server accepts. Valid values: 1 - 1024. Default is 100.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: 0 to 1, 2000 to 1024)

Number of delivery attempts Specifies the number of delivery attempts for a mail. Minimum are two delivery attempts, i.e., after a failed delivery attempt, at least one more attempt must be made. Valid values: 2 - 10. Default is 5.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: 1 to 2, 50 to 10)

Interval between delivery attempts (in minutes) Specifies the interval in minutes between the delivery attempts. Valid values: 1 - 10. Default is 5.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: 0 to 1, 15 to 10).

5.1.3   HTTP server

Offer HTTP Server Specifies whether the HTTP server should be offered. Valid values: `true` or `false`.

Local IP address to bind HTTP Server to Here you can specify the IP address of the network card to which the HTTP server is to be bound.
If left blank, the HTTP server binds to all existing network interfaces. Valid values: any valid IP address or empty.

HTTP port Specifies the HTTP port (default is 80). Valid values: 1 - 65535.

Offer HTTPS Server Specifies whether the "HTTP over SSL" server (HTTPS server) should be offered. Valid values: `true` or `false`.

Local IP address to bind HTTPS Server to Here you can specify the IP address of the network card to which the HTTPS server is to be bound.
If left blank, the HTTPS server binds to all existing network interfaces. Valid values: any valid IP address or empty.

HTTPS port Specifies the HTTPS port (default is 443). Valid values: 1 - 65535.

Path to keystore file To provide the HTTPS server, the path to a keystore file that stores a certificate required for the TLS protocol needs to be specified.
If a relative path is specified, it is interpreted relative to the root directory of the FB-MailService. If the keystore file is in the root directory of the FB-MailService, only the name of the file needs to be specified here.

keystore format Specifies the format of the keystore. Valid values: `JCEKS`, `JKS` or `PKCS12`.

Keystore password Specification of the keystores password.

Fully qualified host name The fully qualified name under which the HTTP server of FB-MailService is reached. Default is `localhost`.

Return files on directory request Specify the files to be returned by the HTTP server when a directory is requested. Several files are separated by a semicolon. The files are searched in the order as stated here. Default is `index.html; index.htm`.
The root folder for files and directories accessible to the HTTP server is the `htdocs` folder in the FB-MailService installation directory.
Note: If the HTTP server is offered, all files in this folder (and any subfolders) can be accessed with the appropriate URL.
All files and folders below the `htdocs` folder must be regarded as **public**!

Offer web service Specifies whether the web service should be offered. Valid values: `true` or `false`. Default is `true`.

Offer web interface Specifies whether the web interface should be offered. Valid values: `true` or `false`. Default is `true`.

Session timeout web service (in minutes) The number of minutes after which a web service session should expire. Permitted data: 1 - 1440. Default is 720, d. H. after 12 hours of inactivity a session expires.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: -10 to 1, 2000 to 1440).

5.1.4   Other settings

Path to mail store Specifies the path to an account / mail folder. If a relative path is specified, it is interpreted relative to the root directory of the FB-MailService. Default is an empty path, ie. the `accounts` folder is created in the root directory of the FB-MailService.
So that the number of hard disk i/os does not exceed the capabilities of the hard disk or the sheer amount of data does not exceed its capacity, any number of paths to account / mail folders can be specified in order to be able to distribute the accounts across multiple hard disks. Simply add another "Path to mail store:" line.
It should be noted that the faster the disk i/os, the more accounts can be stored on the hard disk, the slower the less. So, on very fast hard drives like SSDs (Solid State Disk), which typically have less capacity smaller but more accounts can be stored. On conventional magnetic storage disks, which have a larger capacity, but which are much slower, less but bigger accounts can be stored.

Determine DNS servers automatically Specifies whether the DNS servers to be used should be determined automatically, or whether the DNS servers specified below should be used. Good and correct functioning DNS servers are very important for accurate spam filtering. The faster the DNS requests are, the faster the spam filtering. Valid values: `true` or `false`. Default is `true`.
Note: If you leave the information in "DNS server 1" and / or "DNS server 2" empty, the DNS servers will be determined automatically even if "Determine DNS servers automatically" was set to `false`.

DNS server 1 Specifies the first DNS server to use if the setting in "Determine DNS servers automatically" is set to `false`.

DNS server 2 Specifies the second DNS server to use if the setting in "Determine DNS servers automatically" is set to `false`. This DNS server is used when the first DNS server does not work.

System language Specifies the system language as an ISO-639-1 abbreviation (for example, `de` for German or `en` for English). If a language is specified which is not available, an available languages will be used. Valid values: ISO 639-1 abbreviation of an available language.

Path to keystore file If the clients (POP3 and SMTP) are required to authenticate on an SSL / TLS connection, the path to a keystore file can be specified here in which a suitable certificate is stored.
If a relative path is specified, it is interpreted relative to the root directory of the FB-MailService. If the keystore file is in the root directory of the FB-MailService, only the name of the file needs to be specified here.

keystore format Specifies the format of the keystore. Valid values: `JCEKS`, `JKS` or `PKCS12`.

Keystore password Specification of the keystores password.

Delete mails at Specifies when files of deleted mails should actually be deleted from the hard disk. Since deleting many files can put a lot of strain on the hard drive, it is possible to specify a time in 24-hour format when the deletion should take place (eg at night when there is little activity). Default: empty (that is, the files are deleted immediately).

Enable public address data and groups Specifies whether to enable the public address data and groups feature. If this setting is `true`, address data and groups in the address book can be given the status "public" and will be visible to every account in the same domain. Default: `false`.

Enable public calendar entries Specifies whether to enable the public calendar entries feature. If this setting is `true`, calendar entries in the calendar can be given the status "public" and will then be visible to every account in the same domain. Default: `false`.

Enable public files Specifies whether to enable the the public files feature. If this setting is `true`, files in the file manager can be given the status "public" and will be visible to every account in the same domain. Default: `false`.

Maximum log file size (in megabytes) Specifies the maximum size of the "MailService.log" log file in megabytes before it is renamed to "MailService (old).log" and the "MailService.log" is rewritten. If there is already a file "MailService (old) .log", it will be deleted first. Allowed data: 10 - 1024. Default is 100.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: 5 to 10, 2000 to 1024).

Maximum statistics file size (in megabytes) Specifies the maximum size of the statistics file "Statistics.stat" in megabytes before it is renamed to "Statistics (old).stat" and the file "Statistics.stat" is rewritten. If there is already a file "Statistics (old).stat", it will be deleted first. Allowed data: 10 - 1024. Default is 100.
Note: If the value is outside the permitted range, the value is set to the nearest limit (examples: 5 to 10, 2000 to 1024).

If the configuration file is changed, the FB-MailService must be restarted for the changes to take effect. In the context menu click on "Stop FB-MailService" and afterwards on "Start FB-MailService" or on the console execute the commands `stop` and `start`.

Below is the default configuration file as it is created in the installation directory when the FB-MailService is installed for the first time. If a configuration file is missing at startup, the FB-MailService is started up applying these settings. Comment lines start with a `#`. The character encoding of the configuration file must be **UTF-8**.

```
[POP3 server]
Offer POP3 Server: true
# [DE]: Specifies to which network card the server should be bound;
# leave blank if the server is on all network cards for incoming
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
Local IP address to bind POP3 Server to:
POP3 port: 110
Offer POP3S server: true
# [DE]: Specifies to which network card the server should be bound;
# leave empty if the server is on all network cards for incoming
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
```

```
Local IP address to bind POP3S Server to:
POP3S port: 995
Path to keystore file:
# [DE]: Either JCEKS, JKS or PKCS12
# [EN]: Either JCEKS, JKS or PKCS12
Keystore format:
Keystore password:
Refuse clear-text login commands: false


[SMTP server]
Offer SMTP Server: true
# [DE]: Specifies to which network card the server should be bound;
# leave blank if the server is on all network cards for incoming
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
Local IP address to bind SMTP server to:
SMTP port: 25
Offer SMTPS server: true
# [DE]: Specifies to which network card the server should be bound;
# leave blank if the server is on all NICs inbound
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
Local IP address to bind SMTPS server to:
SMTPS port: 465
Path to keystore file:
# [DE]: Either JCEKS, JKS or PKCS12
# [EN]: Either JCEKS, JKS or PKCS12
Keystore format:
Keystore password:
# [EN]: fully qualified host name of the SMTP server, e.g. For example, mail.example.com
# [EN]: Fully qualified host name of the SMTP server; For example: mail.example.com
Fully qualified host name: localhost
%Relay mails after filtering: false
%Relay mails to:
Maximum mail size (in megabytes): 100
#: TLS setting (encryption) for direct mail delivery:
# Never run STARTTLS or run STARTTLS if available
# 0 = never; 1 = if available
# [TL]: TLS setting for direct mail sending:
# never conduct STARTTLS, or conduct STARTTLS if available
# 0 = never; 1 = if available:
TLS setting: 1
# [DE]: valid dates are between 2 and 10 attempts:
# [EN]: Valid specifications between 2 and 10 attempts:
Number of delivery attempts: 5
# [DE]: Valid dates range from 1 to 10 minutes:
# [EN]: Valid specifications are between 1 and 10 minutes:
Interval between delivery attempts (in minutes): 5


[HTTP Server]
Offer HTTP Server: true
# [DE]: Specifies to which network card the server should be bound;
```

```
# leave empty if the server is on all network cards for incoming
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
Local IP address to bind HTTP Server to:
HTTP port: 80
Offer HTTPS server: true
# [DE]: Specifies to which network card the server should be bound;
# leave blank if the server is on all network cards for incoming
# Connections should wait
# [EN]: Binding specification; empty, if server is to listen to all
# network interface cards:
Local IP address to bind HTTPS server to:
HTTPS port: 443
Path to keystore file:
# [DE]: Either JCEKS, JKS or PKCS12
# [EN]: Either JCEKS, JKS or PKCS12
Keystore format:
Keystore password:
# [DE]: Fully qualified host name of the HTTP server, eg. Example.com
# [EN]: Fully qualified host name of the HTTP server; For example: example.com
Fully qualified host name: localhost
# [DE]: For multiple files with; separate from each other
# [EN]: Separate several files with;
Return files on directory request: index.html; index.htm
Offer web service: true
Offer web interface: true
# [DE]: Valid dates range from 1 to 1440 minutes:
# [EN]: Valid specifications are between 1 and 1440 minutes:
Session timeout web service (in minutes): 720

[Miscellaneous]
Path to mail store:
Determine DNS servers automatically: true
DNS server 1:
DNS server 2:
# [DE]: ISO-639-1-Code zur Angabe der Systemsprache (z. B.
# "de" für deutsch, "en" für englisch, "fr" für französisch):
# [EN]: ISO-639-1 code in order to specify the system language (eg
# "de" for german, "en" for english, "fr" for french):
System language: de
# [DE]: In the event that the FB-MailService as a client joins a
# Authenticate SSL / TLS connection, here can be a keystore
# are given:
# [EN]: In case the clients have authenticated themselves on a SSL / TLS
# connection, a keystore can be specified here
Path to keystore file:
# [DE]: Either JCEKS, JKS or PKCS12
# [EN]: Either JCEKS, JKS or PKCS12
Keystore format:
Keystore password:
# [DE]: Valid dates are between 0:00 and 23:59; leave blank if
# Mail files should be deleted immediately
# [EN]: Valid specifications are between 0:00 and 23:59; leave empty if mail
```

```
# files are to be deleted immediately
Delete mails at:
Enable public address data and groups: false
Enable public calendar entries: false
Enable public files: false
Maximum log file size (in megabytes): 100
Maximum statistics file size (in megabytes): 100
```

5.2 The admin account

The admin account is available from the beginning every time the FB-MailService starts up. Unlike all other accounts, its name consists only of the local mailadress part **postmaster** and the account can not be deleted.

The password of this account is unknown at the beginning, to prevent newly set up FB-MailService installations being vulnerable. To log in for the first time with the admin account, the postmaster password must first be reset (either by the context menu or the console) to get the current password. This password must also be changed when logging in for the first time via the web service.

The domain of this account results from the fully qualified name under which the SMTP server of the FB-MailService is reached. This is specified in the configuration file (default is `localhost`).

The admin account has only the restrictions that it can not be deleted, the name of the account is set to "postmaster", and its domain is the fully qualified host name under which the SMTP server of the FB-MailService is reached. Otherwise, this account can be used like any other account. In addition, however, with the admin account the following web service operations can be called:

- getDomainsAndMailStores (retrieve existing domains and mail stores)

- renameDomain (rename domain)

- addAccount (add account)

- getAccounts (retrieve existing accounts from a domain)

- deleteAccounts (delete accounts)

- renameAccount (Rename an account)

- resetPassword (reset an accounts password)

- setUserMustChangePassword (The setting whether a user must change his password at the next login)

- cancelSessions (cancel all active sessions)

- setStorageQuota (change an accounts storage quota)

- setNumberOfRecipientsAllowedToAddress (change an accounts allowed number of different recipient addresses)

The reason why the account name is set to "postmaster" is due to the SMTP specification. According to RFC 2821, section 4.5.1[4] each SMTP server that supports mail forwarding or mail delivery must have a mailbox with the local name (case-insensitive) "postmaster". This is done with the admin account.

All mail addressed to the mail address `postmaster` or `postmaster@[domain]` will be accepted and delivered to the admin account, where `[domain]` is either the domain of one of the accounts the FB-MailService manages, or the fully qualified host name under which the SMTP server of the FB-MailService is reached.

---

[4]http://tools.ietf.org/html/rfc2821#section-4.5.1

5.3   The Account assistant

Except through the web service operation "addAccount" (called with the admin account) accounts can also be created using the account assistant.
The Account Wizard allows you to create accounts with usernames, aliases, and passwords read from plain text files.
It is opened by the context menu by clicking on the menu item "Account assistant" or the console command `accounts`.
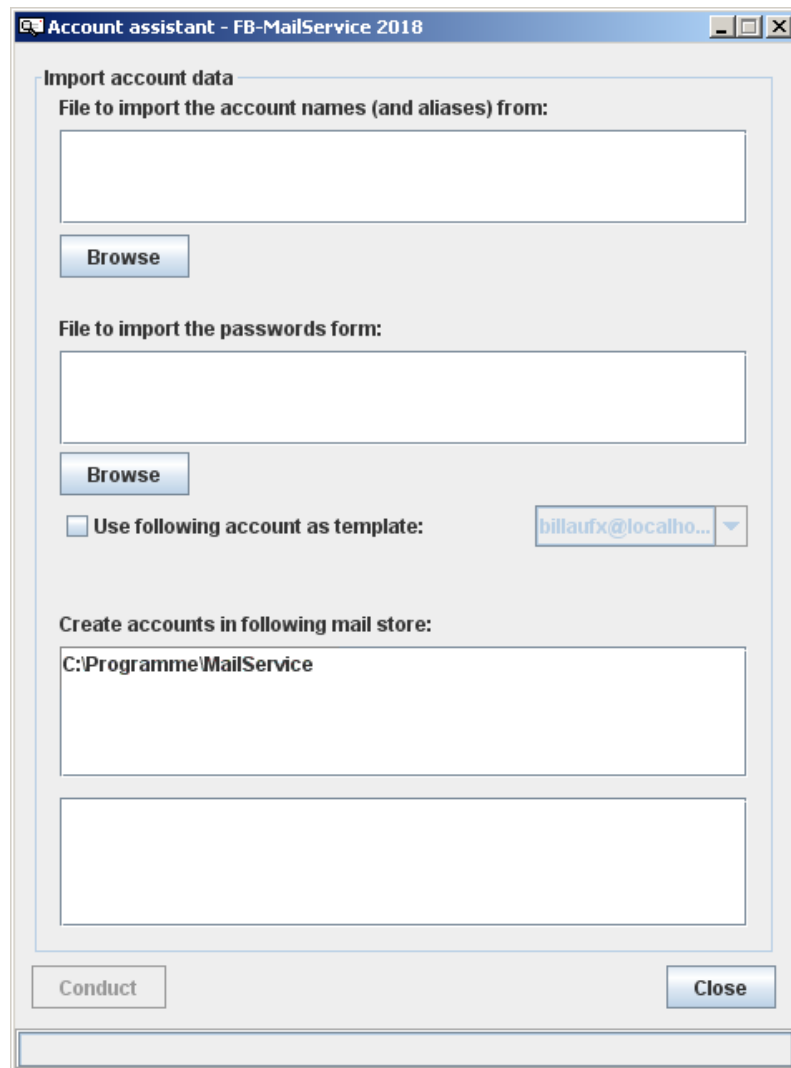
In this dialog, the path to two text files can be specified, from which the account names and passwords for the new accounts to be imported are to be read in.
The account names must each be in a separate line in the file. These account names must be valid email addresses.
If an account has aliases, they must be in this line, separated by tabs. The first entry of a line is interpreted as the account name, all other entries of a line are interpreted as aliases. The account names must be valid mail addresses (ie. no umlauts may occur in the local part of the email address, for example). Aliases can only be the local part of an e-mail address (without @ sign and without the domain).
Separator between the entries of a line is, as just mentioned, the tabulator character:

```
account1@domain1.tld Alias1_1 Alias1_2 Alias1_3
```

```
account2@domain1.tld Alias2_1 Alias2_2
account3@domain2.tld
account4@domain1.tld Alias4_1
...
```

In addition, another file can be specified, from which the passwords can be imported. They must be in plain text (not encrypted) in the exact same order in the password file as the names of their accounts in the account name file.

If no password file is specified, a newly created password is assigned to each read-in account. In this case, a login to each of these imported accounts is only possible, if the current password was previously obtained through the web service call `resetPassword` (only possible using the admin account).

In addition, the mail store must be specified in which the accounts are to be created. Furthermore, an account can serve as a template. The following properties of the template account are transferred to the newly created accounts:

- Whether the user must change the password the next time they log in to the HTTP server (via the web service)

- Whether to save sent mails in the folder `Sent messages`

- Whether the recipient addresses of sent mails should be added to the address book

- The number of allowed recipient addresses per day

- The allocated storage quota

- The SMTP settings

By clicking on the button "Execute" the account names (and aliases) and the passwords are read in and the new accounts are created. Upon completion, it will be informed

- if all aliases could be added to the respective account.

- if all accounts could be added to the account collection.

If not all aliases and / or accounts could be added, the reasons for this can be found in the log file.

Note:

- The character encoding of the account name and password file must be **UTF-8**.

- If an account name or alias already exists, that account will not be added.

- Since the admin account already exists at the first start of the FB-MailService, no account or alias having the name "postmaster" can be added.

- Account name and password must not be empty. If one of them is empty, the account will not be added and the import continues with the next line.

- The account names must be valid mail addresses (ie. no umlauts may occur in the local part of the mail address, for example). If an account name is not a valid email address, the account will not be created.

- The aliases must be valid local parts of an e-mail address (that is, for example, no umlauts may occur in them). If an alias is not a valid local part of an email address, it will not be added to the account.

- If a template account is used and this template account has been configured to require the password to be changed, all logons to the SMTP or POP3 server of the FB-MailService will be rejected for all newly created accounts with an indication.
Only login processes to the HTTP server (via the web service) are possible for these accounts, where only the action "Change password" (or "Log out") can be performed until the password has actually been changed. Only after the password has been changed, you can log on to the SMTP and POP3 server of the FB-MailService using these accounts.

6   Statistics

To evaluate its statistics files, the FB-MailService provides a user interface.
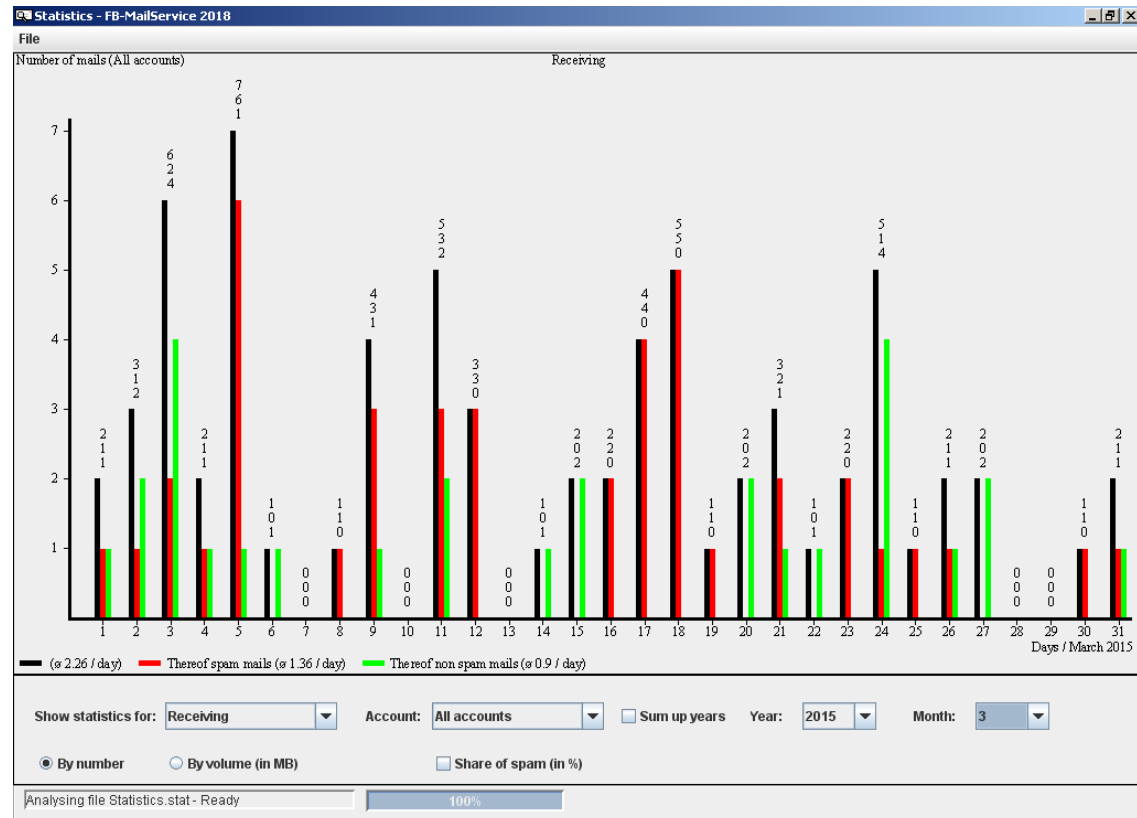
By clicking on the file menu item "Start analysis", the statistics file "Statistics.stat" in the installation directory of the FB-MailService is read in and evaluated by default. By clicking on the file menu item "Start analysis ..." any FB-MailService statistics file can be specified, which should be read in and evaluated.

Depending on the size of the statistics file to be read, the evaluation may take a few seconds. About the progress is informed with a progress bar in the status bar.

The statistics files can be evaluated after

- received mails/sent mails

- for all accounts/for each account

- in total and for years/months/days

- by the number of emails/by the volume of emails (in megabytes)

- by absolute numbers/by the relative percentage of spam mails
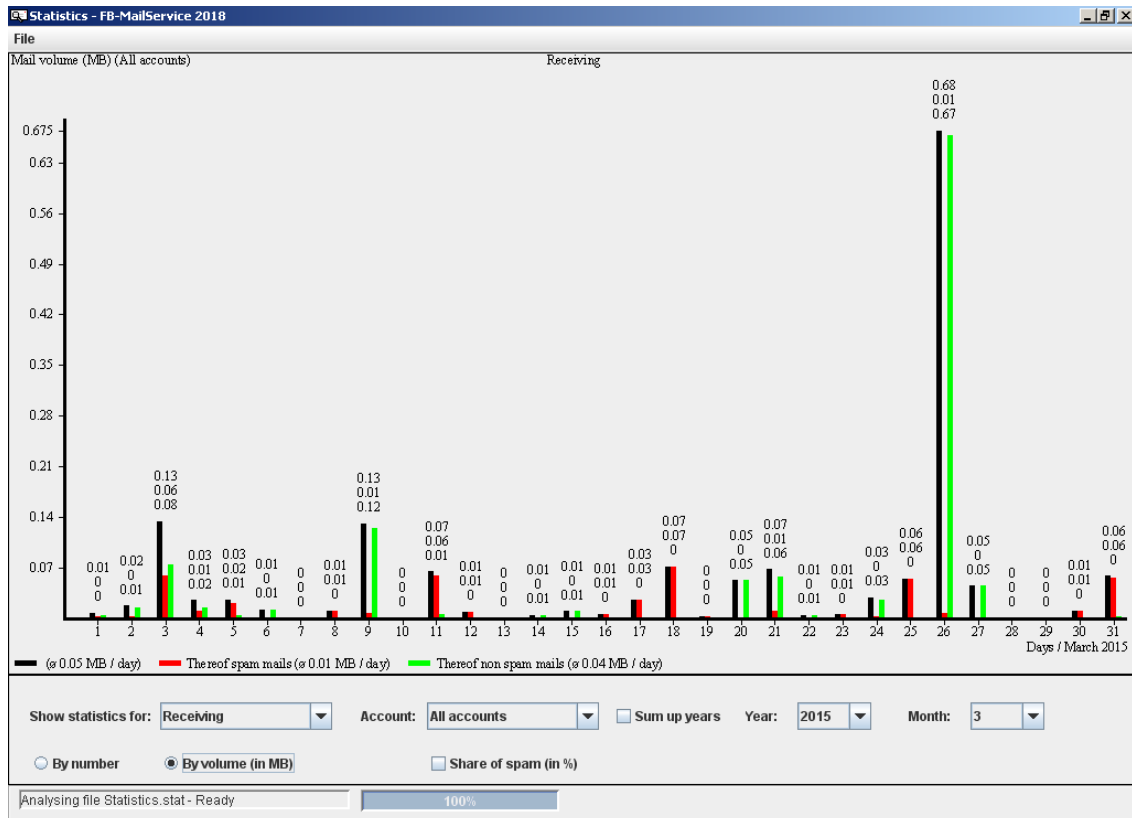
For evaluation by absolute numbers, three different colored bars are displayed:

- Black for all mails (ie. all mails scanned for spam, including all accepted and retrieved mails).
  Note: In the view for sent mails the black bar stands only for all mails sent. Of course, there is no spam check for emails sent by the FB-MailService from authenticated users.

- red for the spam mails it contains

- green for the non-spam mails it contains

In the legend, the average amount per time unit (day, month, year) is shown in brackets.

In addition to the evaluation according to absolute numbers, the evaluations are also available relative to the proportion of spam emails.
Each diagram can be saved as a bitmap or JPEG file by the menu item "Save diagram".
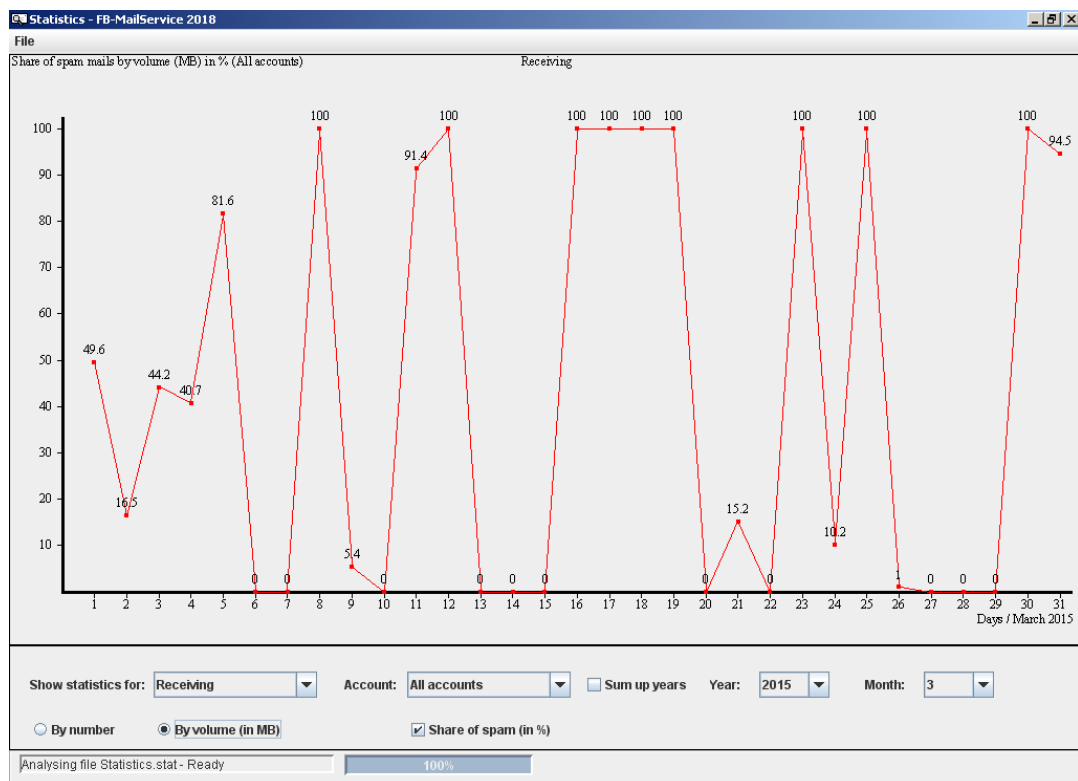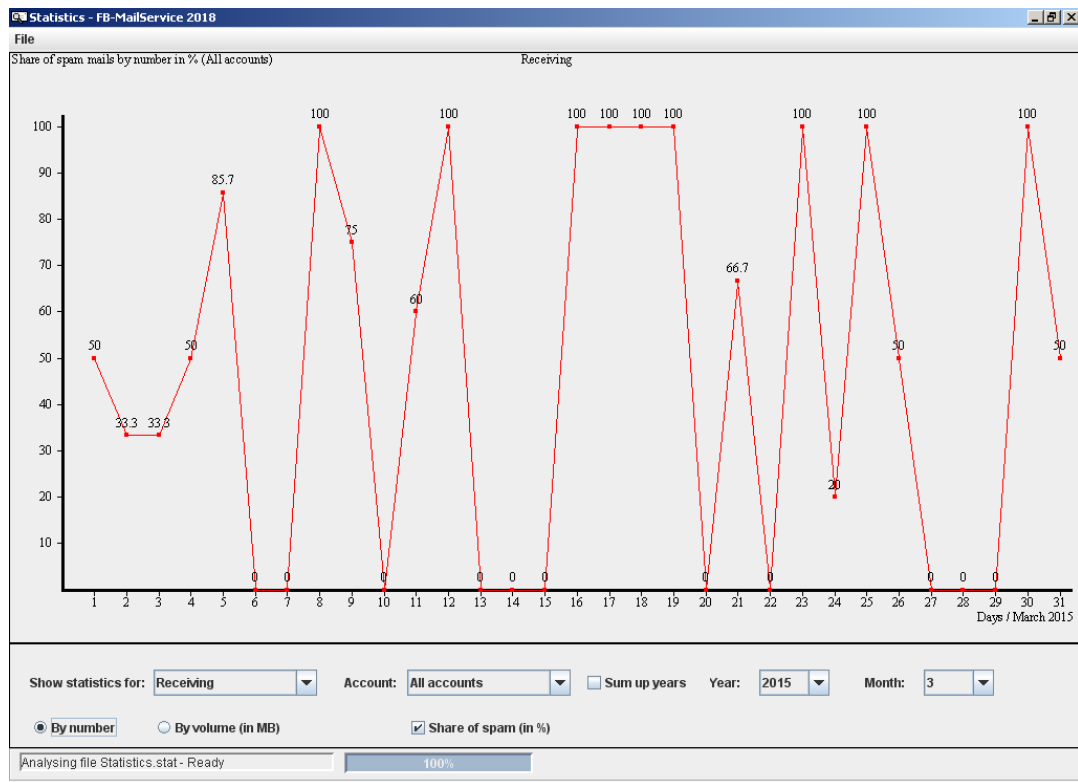
7   The WebServiceClient

The web service of the FB-MailService enables clients to carry out extensive control, administration and configuration of their accounts and complete mail correspondence.
The advantage: Mails, contacts, appointments and files remain on the server. The interface is known and the calls are specified. This makes it possible to have a uniform view of this data from anywhere and with any terminal and client that knows the web service calls. FB Software offers such a client with the WebServiceClient.

It will be described in this section. The specification of the individual web service calls can be found in the appendix at A - Specification of web service calls.

If the executable jar file of the WebServiceClient is in the `htdocs` folder of the FB-MailService (after installation it is by default), the WebServiceClient can be downloaded using the URL `http://[domain]/wsc` where `[domain]` is the name of the computer on which the FB-MailService is running and under which its HTTP server can be reached.
This will allow you to access your mails, addresses, appointments and files from any computer that has a current Java Runtime Environment installed.

The WebServiceClient is started like the installer and the FB-MailService by starting the executable JAR file "WebServiceClient.jar".
After installing the FB-MailService, this file is in its installation directory.
With the start of the WebServiceClient, the login mask also appears. This can also be called up by the menu item "File ⇒ Login".
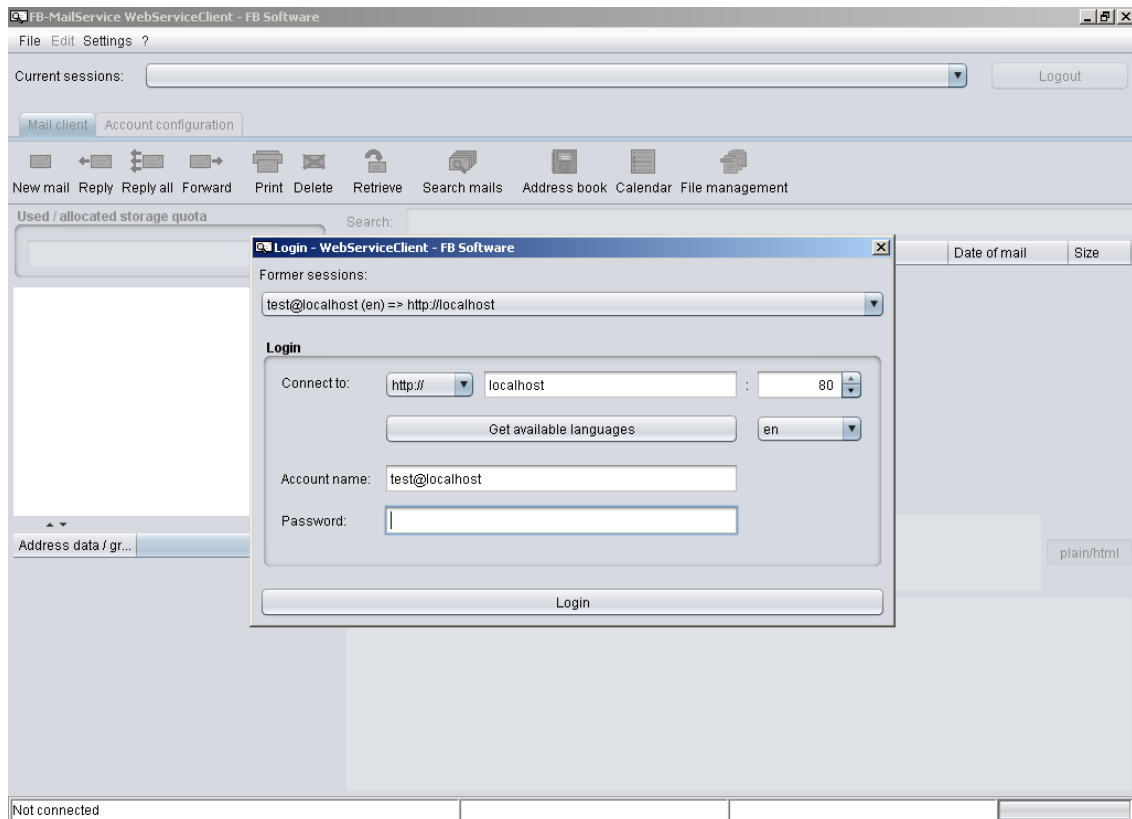
Note: In order to be able to connect to an FB-MailService using the WebServiceClient, this FB-MailService must offer the HTTP or HTTPS server and the web service (to be set in the

configuration file).
In addition, the computer on which the FB-MailService is executed must be reachable for the
WebServiceClient, ie. for example, intervening firewalls must be configured appropriately, etc.
To protect the password of an account when logging in, it is strongly recommended that the web
service call is done only using an encrypted connection, ie. to use the HTTPS protocol.

By the menu item "View ⇒ Language" the language for the user interface can be selected out of
the available languages.

All web service operations offered by FB-MailService can be accessed using the WebServiceClient.

The available languages for the FB-MailService error messages can be retrieved without having to
log in first with an account. For all other calls you must first have logged in with an account.
To log in, you must specify:

- The protocol for the connection (http or https; if the HTTPS server is offered, it is recom-
  mended to use this protocol as the transmitted data will then be encrypted)

- The fully qualified network name of the computer running the FB-MailService to connect to

- The port on which the FB-MailService listens for the specified protocol (default 80 for http
  and 443 for https)

- The name and password of the account you want to log in with

- Optional: A language from the previously retrieved available languages for the FB-MailService
  error messages. If the entry remains empty, the system language of the FB-MailService is
  used for the error messages.

Note: Multiple sessions can be logged in in parallel for the same FB-MailService account (that is,
two sessions for one account are not mutually exclusive), but only from different WebServiceClient
instances. This means that several people can use one and the same FB-MailService account at
the same time, if this is considered appropriate.

From the same WebServiceClient instance, you can log in using as many accounts as you like
at the same time. The logged-in sessions are listed in the "Current sessions" combo box at the
top and can be changed using this combo box.
When the main window of the WebServiceClient is closed, any sessions that are still logged in are
automatically logged out.

The WebServiceClient is divided into two parts: the mail client and the account configuration.

7.1   The mail client

The WebServiceClient offers the usual comfort of a full-fledged mail client. In the upper left area
the available folders are listed. In the upper right area the mails of a folder are listed. In the lower
left area the contacts and groups of the address book are listed, in the lower right area a preview
of the currently selected mail in the mail list is displayed.
By the menu item "Edit ⇒ Edit Templates" the templates for newly created mails can be man-
aged.
Right-clicking on the appropriate buttons ("New Mail", "Reply", "Reply All", "Forward") will list
the available templates. By clicking on one of the templates, this template will be used for the
mail to be created and the template will be set as default, so that subsequent clicks on one of the
buttons automatically use this template.
If the call used to start the WebServiceClient is appended with paths to files to be sent as attach-
ments, the create mail mask is displayed after logging on with the specified files as attachments.
If a path contains spaces, it must be enclosed in quotation marks. Examples:
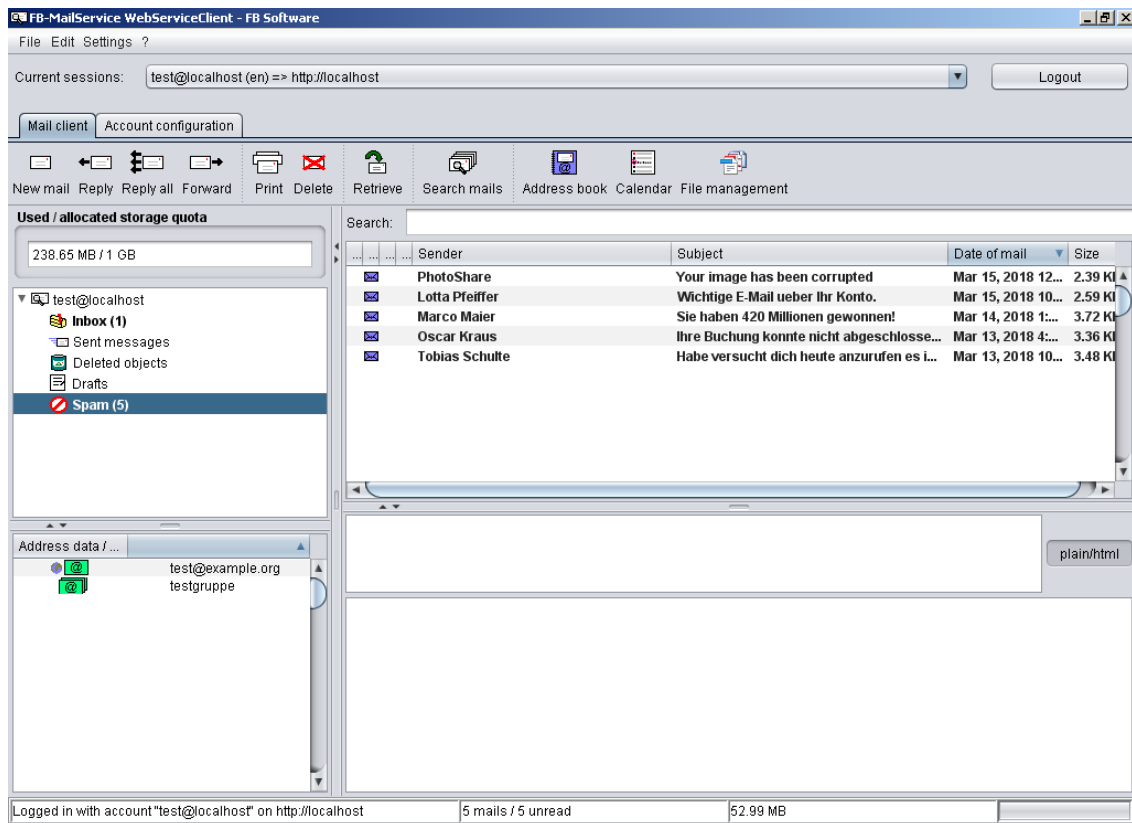
```
java -jar WebServiceClient.jar "C:\path to file\document.pdf" "document 2.pdf"

java -jar WebServiceClient.jar "/home/user/Documents/New Document.pdf"
```

Relative paths are interpreted relative to the root of the WebService client.
If only a single path is specified and the path specifies a mail file, ie ends on `.eml`, the mail file is not interpreted as an attachment to be sent but as a mail file to be opened and displayed accordingly.
If only a single argument is specified, starting with `mailto:`, this argument will be interpreted as a `mailto` URL, the mail creation will be opened, and the elements of the URL (recipient, subject, content) entered in the corresponding fields.

By right-clicking on the button "Retrieve", the configured POP3 accounts are listed. By clicking on a POP3 account, e-mails are retrieved from this POP3 account. By clicking on the button "Retrieve" all configured POP3 accounts are retrieved.
Once retrieved, the new mails will be in the `Inbox` and/or `Spam` folder.

If a mail contains attachments, a large button showing a paperclip icon will appear to the right of the mail preview. Right-clicking this button displays a menu that opens a dialog for saving or uploading the attached files. By left-clicking on this button, the existing attachments of the mail are listed. Click on a file name to open the attached file. When the file names are passed over, a submenu is opened which can be used to open, save or upload the attachment to the file manager. Detected appointments from iCalendar files (file extension ".ics") can also be added to the calendar.

At the top of the mail listing a search box for a quick search is available. This allows the currently listed mails to be searched locally, according to the following fields:

- sender

- sender address

- "To", "CC" and "BCC" fields

- subject

- date

- size specification

For multiple search terms, an AND link is assumed (that is, all search terms must be found for a mail to match the search criteria). Multiple keywords enclosed in quotation marks are interpreted as a search term.

By clicking on the button "Search mails", you can also open the search mask, which allows a more precise specification of the search criteria and, in addition, the message body can be searched. The search placed by the search mask takes place on the server.

By clicking on the button "Address book" the address book management is called up.
Here the contacts and groups of the private address book can be managed and public address data and groups of other accounts of the same domain can be viewed.
Address data in CSV format (comma separated values) can be imported into the private address book. For the values to be assigned, the columns must be as follows:

- `mailAddress`

- `imageFileContentBase64Encoded`
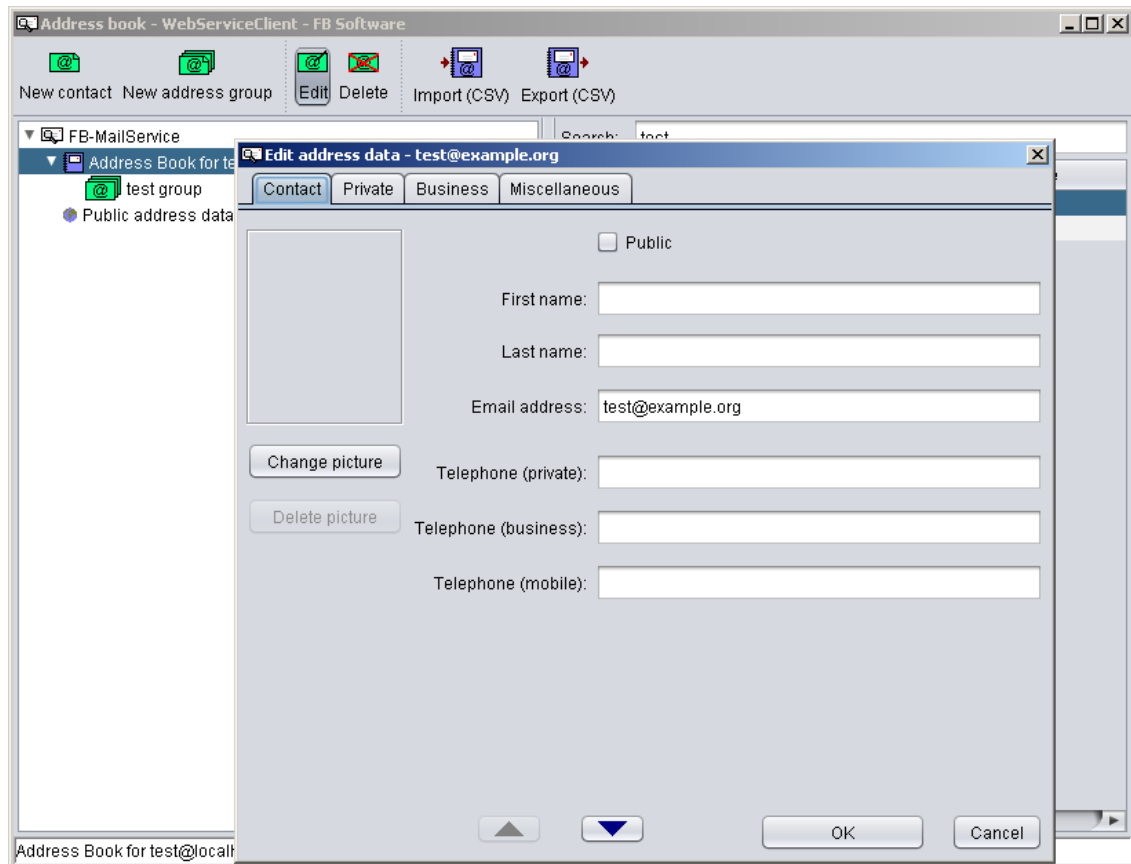
- `firstName`

- `lastName`

- `privateStreet`

- `privateZipCode`

- `privateCity`

- `privateState`

- `privateCountry`

- `privatePhoneNumber`

- `privateFax`

- `privateWebSite`

- `birthday`

- `mobilePhoneNumber`

- `businessStreet`

- `businessZipCode`

- `businessCity`

- `businessState`

- `businessCountry`

- `businessPhoneNumber`

- `businessFax`

- `businessWebSite`

- `role`

- `company`

- `miscellaneous`

There may also be fewer columns. The order of the columns is irrelevant, but only values from columns that have one of the given names can be assigned.
The CSV file is expected to be encoded in UTF-8.

Export function note: If address records are selected, only the selected records of an address book or an address group will be exported. If no records are selected, the entire address book or the entire address group will be exported.

By clicking on the button "Calendar" the calendar management is called up.
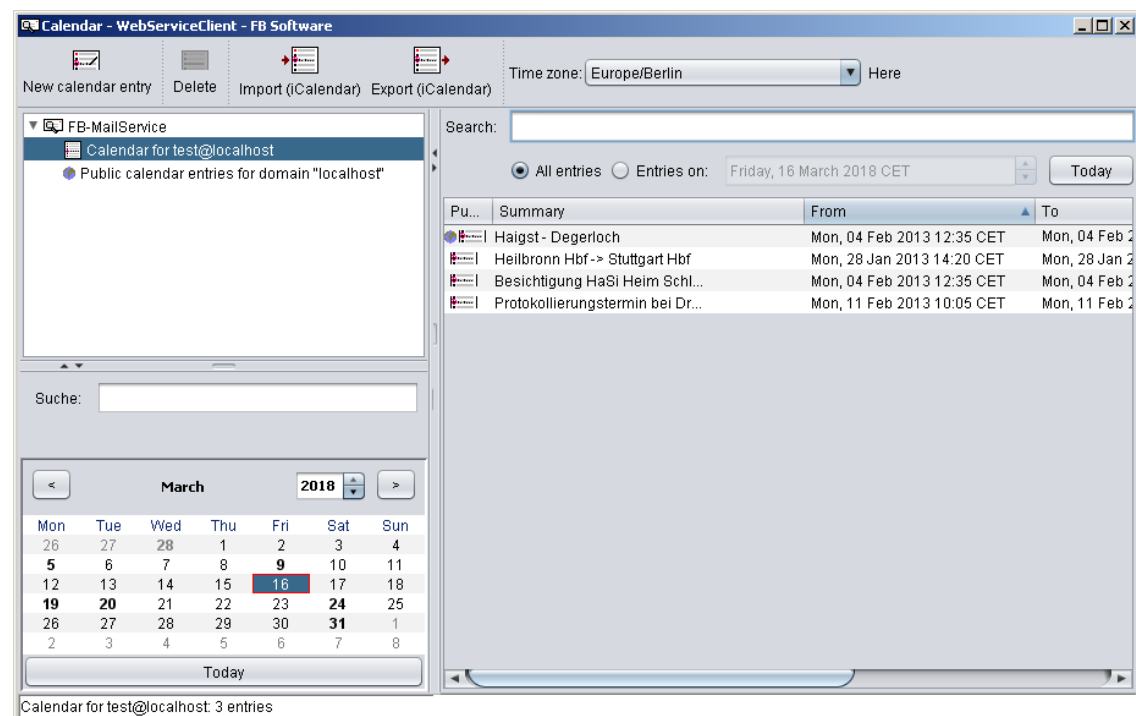
Here, the entries of the private calendar can be managed and public calendar entries of other accounts of the same domain can be viewed.
Calendar entries can be sent by e-mail (in iCalendar format[5]) and appointments received by mail (in iCalendar format) can also be added to the calendar.
The WebServiceClient can remind you of calendar entries. For this purpose, a corresponding

---

[5]http://tools.ietf.org/html/rfc5545

message window opens that overlays all other open windows. If desired, an acoustic signal is also played.

In addition, the FB-MailService can send a reminder mail to all mail addresses registered in the column "reminder by mail to".

Note on the export function: If calendar entries are marked, only the selected entries of a calendar are exported. If no entries are marked, the entire calendar will be exported.

By clicking on the button "File management" the file management is called up.
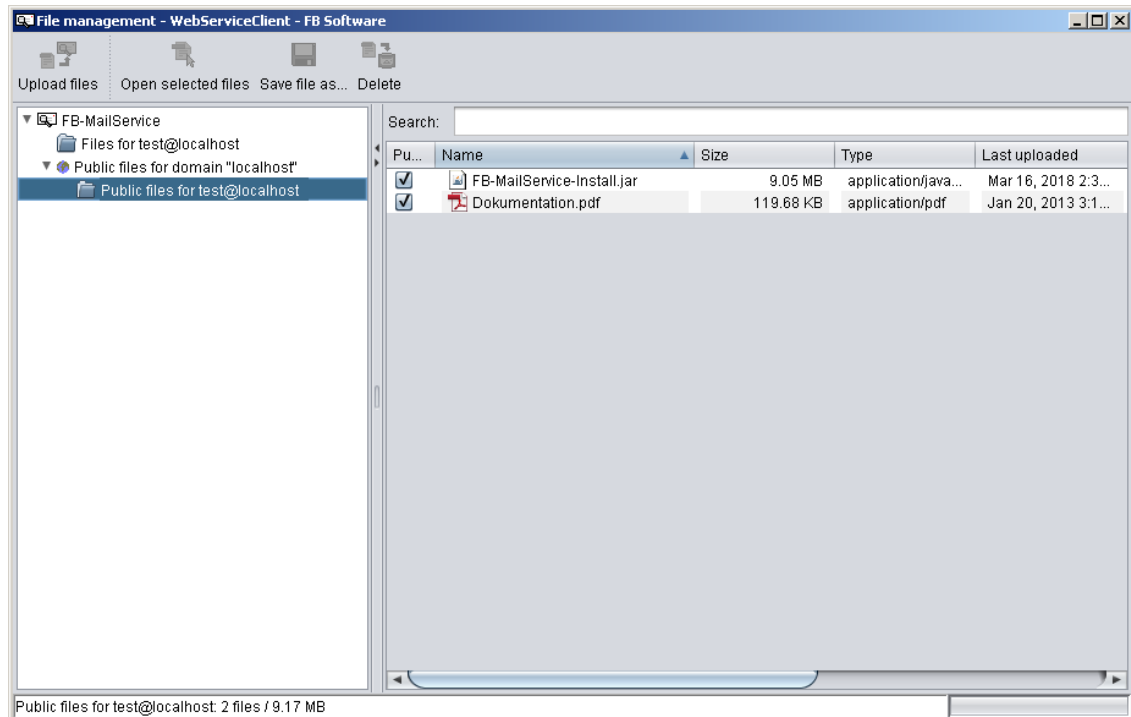
Here files can be uploaded to the server and the public files of other accounts of the same domain can be viewed, opened and saved. For own files the description and the status "public" can be changed.

Both own files and public files of other accounts can be sent by mail attachment.

When logged in with the admin account, the "Account Management" button is visible next to the
"File management" button. Click on this button to open the account management.
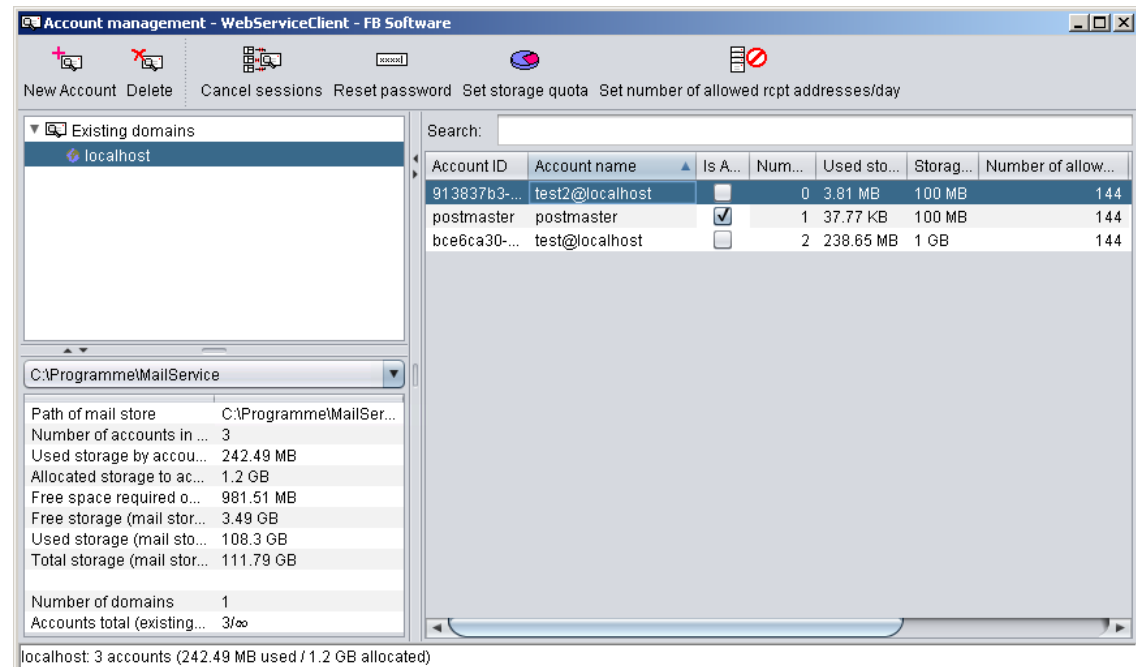
Here you can retrieve existing domains and accounts, reset data structures, abort sessions, reset
passwords, and change the storage quota and the number of allowed recipient addresses per day
and account. You can also create new accounts and delete existing accounts.
Renaming domains and accounts can also be done here.
Furthermore, the property can be changed, if the user has to change the password at the next
login and all accounts can be addressed by mail.

The WebServiceClient uses the "notify" WebService call to be notified of the arrival of new mail
in the `Inbox` and/or `Spam` folder.

Upon receipt of new mails, an audible signal is played (if desired) and a blue unopened enve-
lope appears in the mask at the bottom right. Clicking on this envelope will list the sessions and
folders that have received new mail. Clicking on an entry will switch to this session, change to the
mail client tab and open the folder.
Note: If changes have been made to the account configuration, those changes will be lost if you
switch to another session if the changes have not already been saved.

7.2   The Account Configuration

At the account configuration tab various settings can be made concerning the account, including password changes, aliases, POP3 accounts, SMTP settings, auto respond/forward, and the management of blacklist and whitelist.

8   The ProcessStarter

For Windows operating systems on which the FB-MailService can only be started automatically, if one accepts a constantly open console window, FB Software offers the ProcessStarter.
The ProcessStarter does nothing else but start a new process (possibly with parameters) using the arguments passed to it and then quit.

The working directory of the process to be started is set to the directory in which the ProcessStarter itself was started.

The first argument passed to the ProcessStarter is interpreted as the path to the file to be executed, all other arguments are passed as parameters to the process to be started.
With this ProcessStarter it is now possible as well on the mentioned operating systems to automatically start the FB-MailService without a permanent open console window, for example, when a user logs in.

To do this, the ProcessStarter.exe file must be copied to the same directory in which the executable JAR file of the FB-MailService "MailService.jar" is located.

In the task scheduler of these operating systems, a task can now be created which starts the ProcessStarter for a specific action (eg "User logs on").

On the "Actions" tab click on the button "New ...".

Next to "Action:" you choose "Start program".

In the text field "Program/Script" the path to the ProcessStarter.exe must be entered, for example: `"C:\Program Files (x86)\FB-MailService\ProcessStarter.exe"`. If the path contains spaces, it must be enclosed in quotation marks.

In the text box "Add arguments (optional)" the arguments must be specified. In this case these are the path to the JVM (java.exe), the parameter "-jar" and then the name of the executable jar file of the FB-MailService "MailService.jar", for example the following three arguments :

`"C:\Program Files (x86)\java\jre8\bin\java.exe" -jar MailService.jar`

On the "Trigger" tab the event can be specified, which initiates the execution of the programs specified on the "Actions" tab (eg. "On log in" or "On start").
However, the ProcessStarter can also be used without task scheduling. If invoked without arguments, a console will appear, in which the entire call of the process to be started can be specified (ie. executable plus parameters).
In this way, for example, if you are not logged in as an administrator, the installer will run with admin rights even if you are no administrator:

- Copy the ProcessStarter.exe file to the directory where the executable Installer JAR file is located.

- Right-click on the "ProcessStarter.exe" file and click on "Run as administrator". Thus, all child processes are executed with admin rights, including the installer.

- Specify the path to the JVM (java.exe) in the console analogous to the start of the FB-MailService, then "-jar" and the name of the executable JAR file of the installer:

`"C:\Program Files (x86)\java\jre8\bin\java.exe" -jar FB-MailService-Install.jar`

However, since the context menu does not appear by right-clicking in the console, you must right-click on the title bar of the console and select the menu item "Edit" in order to display the usual menu. Here the entire call can also be inserted by the menu item "Insert" if it was previously copied to the clipboard.
The ProcessStarter can also be used to set up the WebServiceClient as the default mail client, so that the WebServiceClient is started when clicking on a "mailto" link.
It should be noted, however, that every time you click on a "mailto" link, a new WebServiceClient instance is started, even if there is already an instance.

9   Notes

The following notes should be observed for a smooth operation of the FB-MailService:

- The admin account (account name "postmaster") is very important. It must not be deleted by hand **by no means**. Without a readable "AccountData.fbms" file in the admin account folder, none of the other accounts will be available. Even if a new admin account with a new "AccountData.fbms" file was created for this case, all other accounts would be no longer available.

- For security, the FB-MailService allows a maximum of 15 failed login attempts (be it via HTTP, POP3 or SMTP server) from an IP address. Thereafter, this IP address is blocked for 24 h and connections from this IP address are immediately closed by the FB-MailService during this 24 h.

- Note on auto-replying to mails: In principle, only directly received mails from an account are automatically answered (that is, no mails retrieved from POP3 servers), if this has been configured accordingly for the account. The following mails are **not** replied to automatically:

  - Mail classified as spam.
  - Mails having an empty "Return-Path" field in the mail header (that is, mails that are missing a "Return-Path" or are as follows: `Return-Path:  <>` or `Return-Path:  `); this includes own auto-reply mails and own delivery status notification messages that are always sent with an empty return path field in the mail header.

  The recipient address of an auto reply email is determined as follows:

  - If there is a non-empty "Reply To" field in the mail header of the mail to be automatically replied, the address specified there will be used.
  - If the recipient's e-mail address can not be retrieved from the "Reply To" field, the e-mail address from the "From" field of the mail header of the mail to be automatically replied is used.
  - If the recipient's e-mail address can not be retrieved from the "From" field, the e-mail address from the "Return-Path" field of the mail header of the mail to be automatically replied is used.

  If no recipient address for the auto-reply mail could be determined in this way, no such mail will be sent.

- If you expect a high memory consumption of the FB-MailService, it makes sense, to direct the JVM to allocate enough memory right at the start using the options `-Xms[size][unit]` (initial size of the heap) and `-Xmx[size][unit]` (maximum heap size). The same applies to the WebServiceClient when emails having large attachments are often opened or created.
  An indication that too little memory was reserved is an error message like:
  `java.lang.OutOfMemoryError:  Java heap space`.
  For example, to reserve 1700 MB for the heap of the FB-MailService or 500 MB for the WebServiceClient, a call would look like this:

  ```
  java -jar -Xms1700m -Xmx1700m MailService.jar
  java -jar -Xms500m -Xmx500m WebServiceClient.jar
  ```

  NOTE: The space reserved in this way must be contiguous. For this call to work, there must be a 1700 (500) MB contiguous amount of free memory in memory. Otherwise, the JVM will abort with an error message even if more than 1700 (500) MB of physical memory is installed.
  On a 64-bit system (assuming sufficient physical memory), much more memory can be reserved than on a 32-bit system.

- The FB-MailService needs in the installation directory as well as all subdirectories full read and write rights. Of course, this also applies to the mail storage folders. There files and folders must be created and deleted and it must therefore be executed with the appropriate rights.

- During the spam check the FB-MailService needs full access to the internet. Existing firewalls must be configured accordingly. If the Internet access is blocked or limited by a firewall, effective spam filtering is no longer possible.

- To the **Blacklist/Whitelist** problem: If one and the same e-mail address is both in the blacklist and in the whitelist, it should be noted that an e-mail is classified as **not** spam in this case. A whitelist entry of an e-mail address takes precedence over a blacklist entry.

- In order for the FB-MailService to be reachable from other computers, the corresponding ports for POP3, SMTP and HTTP servers (possibly also POP3S, SMTPS, HTTPS) must be opened in each firewall in front of the computer on which the FB- MailService is executed.

- If the FB-MailService shows an unexpected or erroneous behavior, the log file "MailService.log" in the installation directory usually shows more details. This file logs important events. This includes:

  - The start of the FB-MailService
  - Terminating the FB-MailService
  - Requests to the HTTP server
  - Filter results of filtered mails
  - The dispatch (ie. the acceptance and the successful delivery) of mails
  - Failed Delivery Attempts
  - Errors encountered

If any errors occur, this file should contain notes or specific error messages that may help to solve the problem.

A   Specification of web service calls

The web service can be used to call the following operations:

1. Log in

2. Log out

3. Retrieve available languages

4. Retrieve existing domains and mail stores (as admin)

5. Rename domain (as admin)

6. Retrieve existing accounts of a domain (as admin)

7. Add account (as admin)

8. Deleting accounts (as admin)

9. Rename an account (as admin)

10. Reset the password (as admin)

11. The setting whether a user must change the password at the next logon, change (as admin)

12. Cancel all active sessions (as admin)

13. Reset data structures (as admin)

14. Change storage quota (as admin)

15. Change allowed number of different recipient addresses (as admin)

16. Change the password of an account

17. Get the storage quota of an account

18. Get sent emails stored in the `Sent` folder

19. Change whether sent mails should be stored in the `Sent` folder

20. Retrieve whether the recipient addresses of sent mails should be added to the address book

21. Change whether the recipient addresses of sent mails should be added to the address book

22. Get SMTP Settings

23. Change SMTP settings

24. Retrieve existing aliases of an account

25. Add aliases to an account

26. Delete aliases from an account

27. Get the blacklist of an account

28. Get the Whitelist of an account

29. Add email address to a blacklist

30. Add email address to a whitelist

31. Delete mail addresses from a blacklist

32. Delete mail addresses from a whitelist

33. Retrieve all POP3 accounts

34. Retrieve a POP3 account

35. Add POP3 account

36. Change POP3 account

37. Delete POP3 accounts

38. Retrieve mails for a POP3 account

39. Retrieve mails for all POP3 accounts

40. Retrieve existing subfolders of a folder

41. Create folder

42. Rename folder

43. Move folder

44. Delete folder

45. Empty a folder

46. To notify about changes to the account

47. Retrieve mails in a folder

48. Search a folder for mails

49. Delete emails

50. Move emails

51. Copy emails

52. Upload emails to file management

53. Change if an email has been answered

54. Change if a mail has been forwarded

55. Change the status Read/Unread of an e-mail

56. Get Auto Answer/Forward settings

57. Change Auto-Answer/Forward Settings

58. Get public address data

59. Get an address book

60. Add address data to an address book

61. Change address data of an address book

62. Change the mail address of an address record

63. Change the property "public" of address data

64. Delete address data from an address book

65. Add an address group to an address book

66. Rename an address group

67. Change an address group

68. Delete address groups from an address book

69. Retrieve existing templates

70. Add a template

71. Change a template

72. Rename a template

73. Delete templates

74. Retrieve public calendar entries

75. Get a calendar

76. Add a calendar entry

77. Change a calendar entry

78. Change the "public" property of calendar entries

79. Delete calendar entries

80. Get public files

81. Retrieve files

82. Add a file

83. Change a file

84. Change the "public" property of files

85. Copy files

86. Delete files

87. Upload attachments of a mail to file management

88. Add an attachment (in iCalendar format) to the calendar

If the web service is offered, a list of all offered web service calls with links to the corresponding WSDL files (Web Service Description Language WSDL) can be found at the URL http://[domain]/wsdl. The URL of the WSDL files themselves, which contain the specification of the web service calls offered, is http://[domain]/wsdl/[WSDL file], or https://[domain]/wsdl/[WSDL-file] - if the web service is offered and the http server with domain [domain] and the standard ports (HTTP: port 80; HTTPS: Port 443) was configured.

NOTE: If the WSDL file is retrieved using the HTTPS protocol, soap: address location in the WSDL file (the address where the web service can be reached) also points to the HTTPS URL, otherwise to the HTTP URL.

All web service calls offered by FB-MailService are described in detail in the following sections. It should be noted that the following web service calls can only be called when logged in with the admin account (account name "postmaster"):

- getDomainsAndMailStores (retrieve existing domains and mail stores)

- renameDomain (rename domain)

- addAccount (add account)

- getAccounts (retrieve existing accounts from a domain)

- deleteAccounts (deleting accounts)

- renameAccount (Rename an account)

- resetPassword (reset the password)

- setUserMustChangePassword (The setting whether a user must change the password at the next login)

- resetDataStructures

- cancelSessions (cancel all active sessions)

- setStorageQuota (change storage quota)

- setNumberOfRecipientsAllowedToAddress (change the allowed number of different recipient addresses)

A.1   Log in

- **path to WSDL file:**

  - `[domain] /wsdl/login.wsdl`

- **input parameter:**

  - `accountName` (type `string`): The name of the account you want to log in with.
  - `accountPassword` (type `string`): The password of the account you want to log in with.
  - `iso639-1LanguageCode` (type `string`): The ISO-639-1 abbreviation of the language to be used for the error messages. Can be left empty, then the current system language is used. If the specified language is not available, the system language will also be used.

- **return value:**

  - `sessionID` (type `string`): The session ID of the newly created session.
  - `userMustChangePassword` (type `boolean`): Indicates whether the password needs to be changed for this account. If so, an error message will be issued for all web service requests having this session ID except for the calls `changePassword` and `logout` until the password was changed.
    In addition, all logons to the SMTP or POP3 server of the FB-MailService are rejected for this account with a reference to it.
    Only after the password has been changed, you can log on to the SMTP and POP3 server of the FB-MailService using this account.
  - `publicAddressDataEnabled` (type `boolean`): Indicates whether or not the Public Address Data feature has been enabled in the FB-MailService configuration file.
  - `publicCalendarEntriesEnabled` (type `boolean`): Indicates whether or not the Public Calendar Entries feature has been enabled in the FB-MailService configuration file.
  - `publicFilesEnabled` (type `boolean`): Indicates whether or not the public files feature has been enabled in the configuration file of the FB-MailService.

- addRecipientsOfSentMailsToAddressBook (type `boolean`): Whether to add the recipient addresses of sent mails to the private address book.

- saveSentMailsInSentFolder (Type `boolean`): Whether to save sent mails in the `Sent` folder.

- senderAddressOfSMTPAccount (Type `string`): The sender's e-mail address used to send the e-mail through a specific SMTP account. If this field is not empty, it is added to the list of possible sender addresses for sent mails in the WebServiceClient, even if the mails should not be sent using a specific SMTP account but directly.

- alias (type `string`): The aliases assigned to this account. This field can occur any number of times to transfer all assigned aliases (equivalent to an array of type `string`).

- domainName (type `string`): The names of the domains the SMTP server of the FB-MailService is responsible for. This field is only returned when the admin account is logged in and can then occur as often as required to transfer all domains (equivalent to an array of type `string`).

- fullyQualifiedHostNameSMTP (type `string`): The fully qualified computer name under which the SMTP server of the FB-MailService is reached.
  Since it is possible to bind the different servers to their own local IP addresses, this information may be needed for the mailing. The WebServiceClient tries to reach the SMTP server of the FB-MailService both by the computer name with which it has logged on to the HTTP server and using `fullyQualifiedHostNameSMTP`.

- smtpPort (type `string`): The SMTP port. -1 indicates that the SMTP server is not offered by the FB-MailService.

- smtpsPort (type `string`): The SMTPS port. -1 indicates that the SMTPS server is not offered by the FB-MailService.

A.2   Logout

- **path to WSDL file:**

  - [domain] /wsdl/logout.wsdl

- **input parameter:**

  - sessionID (type `string`): The session ID of the session you want to log out of.

- **return value:**

  - loggedOut (type `boolean`): Indicates whether the session has been logged off.

A.3   Get available languages

- **path to WSDL file:**

  - [domain] /wsdl/getAvailableLanguages.wsdl

- **input parameter:**

- **return value:**

  - iso639-1LanguageCode (type `string`): The ISO-639-1 abbreviation of an available language. This field occurs at least once (the system language), and can otherwise occur as often as desired (equivalent to an array of type `string` with at least 1 element).

NOTE: Fetching available languages is provided to allow you to specify the preferred language for any server-side error messages that might occur during the session.

A.4   Retrieve existing domains and mail stores (as admin)

- **path to WSDL file:**

    - `[domain] /wsdl/getDomainsAndMailStores.wsdl`

- **input parameter:**

    - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.

- **return value:**

    - `totalNumberOfExistingAccounts` (type `int`): The number of all existing accounts.
    - `numberOfAccountsCoveredByLicense` (type `string`): The number of accounts covered by the license.
    - `domainName` (type `string`): The name of an existing domain. This field occurs at least once (the fully qualified name of the SMTP server = domain of the admin account) and can otherwise occur as often as required to transfer all domains (equivalent to an array of type `string` with at least 1 element) ,
    - `mailStore` (type `mailStore`): A mail store specified in the configuration file. This field occurs at least once (the default mail store in the root directory of the FB-MailService) and can otherwise occur as often as required to transfer all mail stores (equivalent to an array of type `mailStore` with at least 1 element). The type `mailStore` consists of the following fields:
        * `path` (type `string`): The path of the mail store.
        * `numberOfAccountsInMailStore` (type `int`): The number of accounts this mail store currently contains.
        * `totalUsedStorageInByte` (type `long`): The amount of space used by all accounts in this mail store.
        * `totalAllocatedStorageInByte` (type `long`): The allocated space of all accounts in this mail store.
        * `serviceableFreeSpaceOnMailStorePartitionInByte` (type `long`): The free space of the disk partition where the mail store resides.
        * `totalSpaceOnMailStorePartitionInByte` (type `long`): The total space of the disk partition where the mail store resides.

A.5   Rename a domain (as admin)

- **path to WSDL file:**

    - `[domain] /wsdl/renameDomain.wsdl`

- **input parameter:**

    - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
    - `domainToRename` (type `string`): The name of the domain to be renamed.
    - `newDomainName` (type `string`): The new domain name.

- **return value:**

    - `renamed` (type `boolean`): Indicates that the operation was performed without error.

Note:

- Renaming a domain renames all the accounts in that domain to the new name. ie. changes the domain part of the account name to `newDomainName`.

- The domain of the admin account can not be renamed. Its domain is the fully qualified name of the SMTP server and can only be changed in the config file of the FB-MailService.

A.6 Retrieve existing accounts of a domain (as admin)

- **path to WSDL file:**

  - `[domain] /wsdl/getAccounts.wsdl`

- **input parameter:**

  - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
  - `domainName` (type `string`): The name of the domain whose accounts are to be retrieved.

- **return value:**

  - `mailServiceAccount` (type `mailServiceAccount`): A MailService account. This field can occur any number of times to transfer all MailService accounts (equivalent to an array of type `mailServiceAccount`).
    The type `mailServiceAccount` consists of the following fields:

    * `accountID` (type `string`): The ID of the account. This ID corresponds to the name of the account folder in the folder `accounts` in the root directory of the FB-MailService. This folder is an FB-MailService account. All mails, the address book, the calendar, uploaded files, as well as black and white list of the account are stored in it.
    * `accountName` (type `string`): The name of the account.
    * `mailStorePath` (type `string`): The path of the mail store in which the account is stored.
    * `numberOfSessions` (type `int`): The number of active sessions on this account.
    * `usedStorageInByte` (type `long`): The amount of space currently used by the account.
    * `storageQuotaInByte` (type `long`): The allocated storage quota for this account.
    * `numberOfRecipientsAllowedToAddressPerDay` (type `int`): The number of different recipient addresses that this account is allowed to send mails to per day.
    * `userMustChangePassword` (type `boolean`): Whether the user needs to change the password at the next logon.
    * `alias` (type `string`): An alias for this account. This field can occur any number of times to transfer all assigned aliases (equivalent to an array of type `string`).
    * `isAdminAccount` (type `boolean`): Whether the account is the admin account or not.

A.7 Add an account (as admin)

- **path to WSDL file:**

  - `[domain] /wsdl/addAccount.wsdl`

- **input parameter:**

  - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.

- nameOfNewAccount (type string): The name of the new account to create.
- passwordOfNewAccount (type string): The password of the new account to be created.
- passwordOfNewAccountConfirmation (type string): Confirmation of the password of the newly created account.
- storageQuotaInByte (type long): The storage contingent in bytes to be assigned to this account. Default is 100 MB.
- numberOfRecipientsAllowedToAddressPerDay (type int): The number of different recipient addresses that this account is allowed to send mails to per day. Default is 144.
- userMustChangePassword (type boolean): Specifies that the user must change the password the first time they log in to the HTTP server (using the web service).
- mailStorePath (type string): The path of the mail store in which the account is to be created. Can be left blank - then the account will be created in the first mail store specified in the configuration file.

- **return value:**

  - added (type boolean): Indicates that the operation was performed without error.
  - accountID (type string): The ID of the newly created account (corresponds to the name of the account folder in the folder accounts in the root directory of the FB-MailService).
  - mailStorePath (type string): The mail store path where the account was created.

Note:

- To add an account, you must be logged in with the admin account.

- Account name and password of the new account to create must not be empty.

- Account names must be valid mail addresses. This means that, for example, no umlauts may occur in the local part of the e-mail address.

- The local part "postmaster" is reserved for the admin account and therefore can not be used as an account name.

A.8   Deleting accounts (as admin)

- **path to WSDL file:**

  - [domain] /wsdl/deleteAccounts.wsdl

- **input parameter:**

  - sessionIDAdminAccount (type string): The session ID of the session logged in with the admin account.
  - accountName (type string): The name of an account to be deleted. This field must occur at least once and may otherwise occur as often as required to specify all accounts to be deleted (equivalent to an array of type string with at least 1 element).

- **return value:**

  - deleted (type boolean): Indicates that the operation was performed without error.

Note:

- Deleting existing accounts is only possible with the admin account.

- The admin account can not be deleted.

- Deleting an account is accompanied by the immediate logout of all sessions for that account.

- Deleting an account entails the deletion of the account folder. All mails still stored as well as address data, templates and black and whitelist will be deleted and can not be restored.

A.9 Rename an account (as admin)

- **path to WSDL file:**

    - [domain] /wsdl/renameAccount.wsdl

- **input parameter:**

    - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
    - `accountToRename` (type `string`): The name of the account to be renamed.
    - `newAccountName` (type `string`): The new account name.

- **return value:**

    - `renamed` (type `boolean`): Indicates that the operation was performed without error.

Note:

- To rename an account, you must be logged in with the admin account.

- Account names must be valid mail addresses. This means that, for example, no umlauts may occur in the local part of the e-mail address.

- The local part "postmaster" is reserved for the admin account and therefore can not be used as an account name.

- The admin account (account name "postmaster" can not be renamed).

- Account names must be unique. If there is already an account name or alias that is `newAccountName`, an error message will be returned.

- If the domain part of the account name is changed, the account will be removed from the previous domain and added to the new domain.

A.10 Reset password (as admin)

- **path to WSDL file:**

    - [domain] /wsdl/resetPassword.wsdl

- **input parameter:**

    - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
    - `accountName` (type `string`): The name of the account for which a new password is to be assigned.

- **return value:**

    - `newPassword` (type `string`): The new password assigned to this account.

Note:

- Resetting the password of an account is only possible with the admin account.

- This call generates and returns a new password. The user must log in using this password the next time.

- The field `userMustChangePassword` is set to `true`, ie. the user must first change the password again directly after the next logon process via the web service, before he can carry out further actions.

- The password of the admin account can not be reset in this way. Otherwise, there is a risk that the admin locks himself out. If the password of the admin account is to be changed, the call `changePassword` must be used, which can be called from the WSC tab "Account configuration", sub-tab "General configuration" in the panel "Change password".

A.11   Change if a user needs to change the password at the next logon (as admin)

- **path to WSDL file:**

  - `[domain] /wsdl/setUserMustChangePassword.wsdl`

- **input parameter:**

  - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
  - `accountName` (type `string`): The name of an account whose setting you want to change. This field must occur at least once and can occur any number of times to specify all accounts whose setting is to be changed (corresponds to an array of type `string` with at least 1 element).
  - `mustChange` (type `boolean`): Specifies whether the user must change the password at the next login or not.

- **return value:**

  - `set` (type `boolean`): Indicates that the operation was performed without error.

A.12   Reset data structures (as admin)

- **path to WSDL file:**

  - `[domain] /wsdl/resetDataStructures.wsdl`

- **input parameter:**

  - `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.
  - `resetMails` (type `boolean`): Whether to reset the data structures in which the mail folders and mails are stored.
  - `resetAddressBook` (type `boolean`): Whether to reset the data structures of the address book.
  - `resetCalendar` (type `boolean`): Whether to reset the data structures of the calendar.
  - `resetFiles` (type `boolean`): Whether to reset the data structures for the file management files.
  - `resetTemplates` (type `boolean`): Whether to reset the data structures for the mail templates.

- accountName (type `string`): The name of an account whose data structures should be reset. This field must occur at least once and can occur any number of times to specify all accounts whose data structures are to be reset (corresponds to an array of type `string` with at least 1 element).

- **return value:**

  - reset (type `boolean`): Indicates that the operation was performed without error.

Note:

- If the data structures in which the mail folders and mails are stored are reset while one or more sessions are active on that account, the data structures are immediately rebuilt. This will change the folder IDs.

- Users of active sessions will notice this by the error message "Invalid Folder ID" when loading a mail folder or mail. In this case, the root mail folder must be reloaded (call context menu for this folder ⇒ "Reload folder").

A.13   Cancel all active sessions (as admin)

- **path to WSDL file:**

  - [domain] /wsdl/cancelSessions.wsdl

- **input parameter:**

  - sessionIDAdminAccount (type `string`): The session ID of the session logged in with the admin account.

  - accountName (type `string`): The name of an account whose sessions you want to cancel. This field must occur at least once and can occur any number of times to specify all accounts whose sessions are to be aborted (equivalent to an array of type `string` with at least 1 element).

- **return value:**

  - canceled (type `boolean`): Indicates that the operation was performed without error.

A.14   Changing the storage quota (as admin)

- **path to WSDL file:**

  - [domain] /wsdl/setStorageQuota.wsdl

- **input parameter:**

  - sessionIDAdminAccount (type `string`): The session ID of the session logged in with the admin account.

  - accountName (type `string`): The name of an account whose storage quota is to be changed. This field must occur at least once and can occur any number of times to specify all accounts whose storage quotas are to be changed (corresponds to an array of type `string` with at least 1 element).

  - storageQuotaInByte (type `long`): The storage quota in bytes to allocate to the accounts.

- **return value:**

  - set (type `boolean`): Indicates that the operation was performed without error.

A.15   Change the allowed number of different recipient addresses per day (as admin)

- **path to WSDL file:**

  – `[domain] /wsdl/setNumberOfRecipientsAllowedToAddress.wsdl`

- **input parameter:**

  – `sessionIDAdminAccount` (type `string`): The session ID of the session logged in with the admin account.

  – `accountName` (type `string`): The name of an account whose number of allowed recipient addresses should be changed per day. This field must occur at least once and can occur any number of times to specify all accounts whose number of allowed recipient addresses per day are to be changed (corresponds to an array of type `string` with at least 1 element).

  – `numberOfRecipientsAllowedToAddressPerDay` (type `int`): The number of different recipient addresses that an account can send mail to per day.

- **return value:**

  – `set` (type `boolean`): Indicates that the operation was performed without error.

A.16   Changing the password of an account

- **path to WSDL file:**

  – `[domain] /wsdl/changePassword.wsdl`

- **input parameter:**

  – `sessionID` (type `string`): The session ID of the session used to log in with the account whose password is to be changed.

  – `oldAccountPassword` (type `string`): The previous password of the account whose password is to be changed.

  – `newAccountPassword` (type `string`): The new password of the account whose password is to be changed.

  – `newAccountPasswordConfirmation` (Type `string`): The confirmation of the new password of the account whose password is to be changed.

- **return value:**

  – `changed` (type `boolean`): Indicates that the operation was performed without error.

Note:

- The new password must not be empty and must not be the same as the previous password.

- Changing the password affects all clients logging on to the FB-MailService for this account (POP3 clients, SMTP clients, web clients).
  If this password is changed, this change must therefore be retraced accordingly in all clients.

A.17 Retrieve storage quota for an account

- **path to WSDL file:**

  – [domain] /wsdl/getStorageQuota.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose storage quota is to be retrieved.

- **return value:**

  – usedStorageInByte (type long): Currently used by the account in bytes.

  – storageQuotaInByte (type long): Allocated storage quota in bytes, ie. the maximum amount of bytes the account may occupy.

A.18 Retrieve whether to save sent mails in the Sent folder

- **path to WSDL file:**

  – [domain] /wsdl/getSaveSentMailsInSentFolder.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose settings are to be retrieved.

- **return value:**

  – save (type boolean): Indicates whether or not sent mails from this account will be stored in the Sent folder of the account.

A.19 Change whether to save sent mails in the Sent folder

- **path to WSDL file:**

  – [domain] /wsdl/setSaveSentMailsInSentFolder.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose settings are to be changed.

  – save (type boolean): Specifies whether to save sent mails from this account in the Sent folder of the account.

- **return value:**

  – set (type boolean): Indicates whether the setting has been set.

A.20 Retrieve whether the recipient addresses of sent mails are added to the address book

- **path to WSDL file:**

  – [domain] /wsdl/getAddRecipientsOfSentMailsToAddressBook.wsdl

- **input parameter:**

– **sessionID** (type **string**): The session ID of the session used to log in with the account whose settings are to be retrieved.

- **return value:**

    – **add** (type **boolean**): Specifies whether the recipient addresses of sent mails are added to the account's private address book.

A.21   Change whether the recipient addresses of sent mails are to be added to the address book

- **path to WSDL file:**

    – [domain] /wsdl/setSaveSentMailsInSentFolder.wsdl

- **input parameter:**

    – **sessionID** (type **string**): The session ID of the session used to log in with the account whose settings you want to change.
    – **add** (type **boolean**): Specifies whether to add the recipient addresses of sent mails to the account's private address book.

- **return value:**

    – **set** (type **boolean**): Indicates whether the setting has been set.

A.22   Retrieve SMTP Settings

- **path to WSDL file:**

    – [domain] /wsdl/getSMTPSettings.wsdl

- **input parameter:**

    – **sessionID** (type **string**): The session ID of the session used to log in with the account for which the SMTP settings are to be retrieved.

- **return value:**

    – **useSMTPAccount** (type **boolean**): Indicates that an SMTP account should be used for sending mail.
    – **smtpServer** (type **string**): The SMTP server over which the mail should be sent if an SMTP account is to be used.
    – **smtpPort** (type **portRange**): The port of the SMTP server through which the mail should be sent if a specific SMTP account is to be used. The type **portRange** is of type **int** in the range 1 - 65535.
    – **isSMTPSPort** (type **boolean**): Indicates whether the port **smtpPort** is an SMTPS port or not.
    – **senderAddress** (type **string**): The sender address to use when sending mail through this SMTP account. Can be left blank if the SMTP server **smtpServer** does not require a specific sender address.
    – **smtpServerNeedsAuthentication** (type **boolean**): Indicates whether the SMTP server requires **smtpServer** authentication.
    – **smtpUser** (type **string**): The SMTP user name, if required.
    – **smtpPassword** (type **string**): The SMTP password if authentication is required.

- noInsecurePasswordTransmission (type boolean): Indicates that the password should not be transmitted unencrypted if authentication is required.
- encryptNever (type boolean): Indicates that the connection to SMTP server smtpServer should never be encrypted. The SMTP port smtpPort must **not** be an SMTPS port.
- encryptIfAvailable (type boolean): Indicates that the connection to SMTP server smtpServer should be encrypted if encryption is available (by the STARTTLS command). The SMTP port smtpPort must **not** be an SMTPS port.
- encryptAlways (type boolean): Indicates that the connection to SMTP server smtpServer must always be encrypted (by STARTTLS command or SMTPS port). The SMTP port smtpPort **may** be an SMTPS port (then it must be marked as such).

A.23   Change SMTP Settings

- **path to WSDL file:**
  - [domain] /wsdl/setSMTPSettings.wsdl
- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which the SMTP settings are to be changed.
  - useSMTPAccount (type boolean): Specifies whether a specific SMTP account should be used for sending mails.
  - smtpServer (type string): The SMTP server over which the mail should be sent if an SMTP account is to be used.
  - smtpPort (type portRange): The port of the SMTP server over which the mail should be sent if an SMTP account is to be used. The type portRange is of type int in the range 1 - 65535.
  - isSMTPSPort (type boolean): Indicates whether the port smtpPort is an SMTPS port or not.
  - senderAddress (type string): The sender address to use when sending mail through this SMTP account. Can be left blank if the SMTP server smtpServer does not require a specific sender address.
  - smtpServerNeedsAuthentication (type boolean): Indicates whether the SMTP server requires smtpServer authentication.
  - smtpUser (type string): The SMTP user name, if required.
  - smtpPassword (type string): The SMTP password if authentication is required.
  - noInsecurePasswordTransmission (type boolean): Specifies whether the password should not be transmitted unencrypted if authentication is required.
  - encryptNever (type boolean): Specifies whether the connection to SMTP server smtpServer should never be encrypted. The SMTP port smtpPort must **not** be an SMTPS port.
  - encryptIfAvailable (type boolean): Specifies whether to encrypt the connection to SMTP server smtpServer if encryption is available (by the STARTTLS command). The SMTP port smtpPort must **not** be an SMTPS port.
  - encryptAlways (type boolean): Specifies whether the connection to SMTP server smtpServer must always be encrypted (by STARTTLS command or SMTPS port). The SMTP port smtpPort **may** be an SMTPS port (then it must be marked as such).
- **return value:**
  - set (type boolean): Indicates whether the SMTP settings have been changed.

Note: Mail addressed to a domain owned by the FB-MailService SMTP server is delivered directly to the local mailbox, even if useSMTPAccount is set to true.

A.24   Retrieve aliases of an account

- **path to WSDL file:**

    – [domain] /wsdl/getAliases.wsdl

- **input parameter:**

    – sessionID (type string): The session ID of the session used to log in with the account whose aliases are to be retrieved.

- **return value:**

    – alias (type string): An alias for this account. This field can occur any number of times to transfer all assigned aliases (equivalent to an array of type string).

A.25   Add aliases to an account

- **path to WSDL file:**

    – [domain] /wsdl/addAlias.wsdl

- **input parameter:**

    – sessionID (type string): The session ID of the session used to log in with the account to which an alias is to be added.
    – aliasToAdd (type string): The alias to add.

- **return value:**

    – added (type boolean): Indicates whether the alias has been added to the account.

Note:

- Aliases must be valid local parts of an e-mail address (that is, for example, no umlauts may occur in it).

- aliases must be unique. If there is already an account name or an alias of this name (within the domain of the account), an error message will be returned.

- The local part "postmaster" is reserved for the admin account and therefore can not be used as an alias.

- The number of possible aliases per account is not artificially limited.

A.26   Delete aliases of an account

- **path to WSDL file:**

    – [domain] /wsdl/deleteAliases.wsdl

- **input parameter:**

    – sessionID (type string): The session ID of the session used to log in with the account from which the aliases are to be deleted.
    – aliasToDelete (type string): An alias for this account to be deleted. This field must occur at least once and can occur any number of times to transfer all aliases to be deleted (equivalent to an array of type string with at least 1 element).

- **return value:**

    – deleted (type boolean): Indicates that the operation was performed without error.

A.27   Retrieve blacklist of an account

- **path to WSDL file:**

  – [domain] /wsdl/getBlacklist.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose blacklist is to be retrieved.

- **return value:**

  – address (type string): An address that is in the blacklist of this account. This field can occur any number of times to transfer all addresses of the list (corresponds to an array of type string).

A.28   Retrieve whitelist of an account

- **path to WSDL file:**

  – [domain] /wsdl/getWhitelist.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose whitelist is to be retrieved.

- **return value:**

  – address (type string): An address that is in the whitelist of this account. This field can occur any number of times to transfer all addresses of the list (corresponds to an array of type string).

A.29   Add email address to blacklist

- **path to WSDL file:**

  – [domain] /wsdl/addMailAddressToBlacklist.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose blacklist an address is to be added to.

  – mailAddressToAdd (type string): The mail address to be added to the accounts blacklist.

- **return value:**

  – added (type boolean): Indicates whether the address has been added to the blacklist.

Note:

- If an e-mail address to be added is already in the blacklist, the e-mail address will not be added and false will be returned.

- If the address is invalid, the address will not be added to the list and an error message will be returned.

A.30   Add email address to whitelist

- **path to WSDL file:**
  - [domain] /wsdl/addMailAddressToWhitelist.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account whose whitelist an address is to be added to.
  - mailAddressToAdd (type string): The mail address to be added to the whitelist of the account.

- **return value:**
  - added (type boolean): Indicates whether the address has been added to the whitelist.

Note:

- If an e-mail address to be added is already in the whitelist, the e-mail address will not be added and false will be returned.

- If the address is invalid, the address will not be added to the list and an error message will be returned.

A.31   Delete mail addresses from the blacklist

- **path to WSDL file:**
  - [domain] /wsdl/deleteMailAddressesFromBlacklist.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account whose blacklist addresses are to be deleted from.
  - mailAddressToDelete (type string): An email address to be deleted from the accounts blacklist. This field must occur at least once and may otherwise occur as often as required to transfer all addresses to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**
  - deleted (type boolean): Indicates that the operation was performed without error.

A.32   Delete mail addresses from whitelist

- **path to WSDL file:**
  - [domain] /wsdl/deleteMailAddressesFromWhitelist.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account whose whitelist addresses are to be deleted from.
  - mailAddressToDelete (type string): An e-mail address to be deleted from the accounts whitelist. This field must occur at least once and may otherwise occur as often as required to transfer all addresses to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**
  - deleted (type boolean): Indicates that the operation was performed without error.

A.33   Retrieve all POP3 accounts of a MailService account

- **path to WSDL file:**

  – [domain] /wsdl/getAllPOP3Accounts.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account whose POP3 accounts are to be retrieved.

- **return value:**

  – pop3Account (type pop3Account): A POP3 account. This field can occur any number of times to transfer all POP3 accounts (corresponds to an array of type pop3Account). The type pop3Account consists of the following fields:

    * pop3Server (type string): The server of the POP3 account.
    * pop3User (type string): The username of the POP3 account used to authenticate to the server pop3Server.
    * basicAddress (type string): The root address of the POP3 account.
    * pop3AccountID (type int): The ID of the POP3 account. This ID is used to reference the POP3 account.

A.34   Retrieve data of a POP3 account

- **path to WSDL file:**

  – [domain] /wsdl/getPOP3Account.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which the data for one of its POP3 accounts is to be retrieved.

  – pop3AccountID (type int): The ID of the POP3 account whose data you want to retrieve.

- **return value:**

  – pop3Server (type string): The name of the POP3 server from which to retrieve mail for this POP3 account.

  – pop3Port (type portRange): The port of the POP3 server from which to retrieve mail for this POP3 account. The type portRange is of type int in the range 1 - 65535.

  – isPOP3SPort (type boolean): Indicates whether the port pop3Port is a POP3S port or not.

  – timeoutInMinutes (type timeoutRange): Indicates how long to wait for a response from the server. If a response from the server is longer than the time specified here, the mail request for this POP3 account will be aborted. The type timeoutRange is of type int in the range of 1 - 10.

  – pop3User (type string): The username to authenticate to the specified POP3 server pop3Server.

  – pop3Password (type string): The password used to authenticate to the specified POP3 server pop3Server.

- noInsecurePasswordTransmission (type boolean): Indicates that no unsecure password transfer should occur. This option can be used to ensure that the password is transmitted encrypted only using a secure method. If there is no secure connection to the server and the server does not offer any of the implemented methods for secure password transmission (CRAM-MD5, APOP), there will be no mail retrieving for this POP3 account, if this option is set.

- mailRetrievingIntervalInMinutes (type intervalRange): Indicates at which minute interval for this POP3 account to periodically retrieve and filter mail from the server. The type intervalRange is of type int in the range of 0 - 1440. A value of 0 means that no automatic polling takes place.

- encryptNever (type boolean): Indicates that the connection should never be encrypted. The POP3 port pop3Port must **not** be a POP3S port.

- encryptIfAvailable (type boolean): Indicates that the connection should be encrypted if encryption is available. The POP3 port pop3Port must **not** be a POP3S port.

- encryptAlways (type boolean): Indicates that the connection must always be encrypted. The POP3 port pop3Port **may** be a POP3S port (then it must be marked as such).

- basicAddress (type string): The root address of this POP3 account.

A.35   Add a POP3 account

- **path to WSDL file:**

  - [domain] /wsdl/addPOP3Account.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account to which a POP3 account is to be added.

  - pop3Server (type string): The name of the POP3 server from which to retrieve mail for this POP3 account.

  - pop3Port (type portRange): The port of the POP3 server from which to retrieve mail for this POP3 account. The type portRange is of type int in the range 1 - 65535.

  - isPOP3SPort (type boolean): Specifies whether the port pop3Port is a POP3S port or not.

  - timeoutInMinutes (type timeoutRange): Specifies how long to wait for a response from the server. If a response from the server is longer than the time specified here, the mail request for this POP3 account will be aborted. The type timeoutRange is of type int in the range of 1 - 10.

  - pop3User (type string): The username to authenticate to the specified POP3 server pop3Server.

  - pop3Password (type string): The password used to authenticate to the specified POP3 server pop3Server.

  - noInsecurePasswordTransmission (type boolean): Specifies that no unsecure password transfer should take place. This option can be used to ensure that the password is transmitted encrypted only using a secure method. If there is no secure connection to the server and the server does not offer any of the implemented methods for secure password transmission (CRAM-MD5, APOP), there will be no mail retrieving for this POP3 account, if this option is set.

- mailRetrievingIntervalInMinutes (type intervalRange): Specifies the minutes interval for this POP3 account to periodically retrieve and filter emails from the server. The type intervalRange is of type int in the range of 0 - 1440. A value of 0 means that no automatic polling takes place.

- encryptNever (type boolean): Specifies that the connection should never be encrypted. The POP3 port pop3Port must **not** be a POP3S port.

- encryptIfAvailable (type boolean): Specifies that the connection should be encrypted if encryption is available. The POP3 port pop3Port must **not** be a POP3S port.

- encryptAlways (type boolean): Specify that the connection must always be encrypted. The POP3 port pop3Port **may** be a POP3S port (then it must be marked as such using the isPOP3SPort option).

- basicAddress (type string): The root address of this POP3 account.

- **return value:**

  - added (type boolean): Indicates whether the POP3 account has been added.

  - newPOP3AccountID (type int): The ID of the newly added POP3 account.

A.36 Change a POP3 account

- **path to WSDL file:**

  - [domain] /wsdl/setPOP3Account.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account to which a POP3 account is to be changed.

  - pop3AccountToSetID (type int): The ID of the POP3 account to be changed.

  - pop3Server (type string): The name of the POP3 server from which to retrieve mail for this POP3 account.

  - pop3Port (type portRange): The port of the POP3 server from which to retrieve mail for this POP3 account. The type portRange is of type int in the range 1 - 65535.

  - isPOP3SPort (type boolean): Specifies whether the port pop3Port is a POP3S port or not.

  - timeoutInMinutes (type timeoutRange): Specifies how long to wait for a response from the server. If a response from the server is longer than the time specified here, the mail request for this POP3 account will be aborted. The type timeoutRange is of type int in the range 1-10.

  - pop3User (type string): The username to authenticate to the specified POP3 server pop3Server.

  - pop3Password (type string): The password used to authenticate to the specified POP3 server pop3Server.

  - noInsecurePasswordTransmission (type boolean): Specifies that no insecure password transfer should take place. This option can be used to ensure that the password is transmitted encrypted only using a secure method. If there is no secure connection to the server and the server does not offer any of the implemented methods for secure password transmission (CRAM-MD5, APOP), there will be no mail retrieving for this POP3 account, if this option is set.

- mailRetrievingIntervalInMinutes (type `intervalRange`): Specifies the minutes interval for this POP3 account to periodically retrieve and filter emails from the server. The type `intervalRange` is of type `int` in the range of 0 - 1440. A value of 0 means that no automatic polling takes place.
- encryptNever (type `boolean`): Specifies that the connection should never be encrypted. The POP3 port `pop3Port` must **not** be a POP3S port.
- encryptIfAvailable (type `boolean`): Specifies that the connection should be encrypted if encryption is available. The POP3 port `pop3Port` must **not** be a POP3S port.
- encryptAlways (type `boolean`): Specify that the connection must always be encrypted. The POP3 port `pop3Port` **may** be a POP3S port (then it must be marked as such using the `isPOP3SPort` option).
- basicAddress (type `string`): The root address of this POP3 account.

- **return value:**

  - set (type `boolean`): Indicates whether the POP3 account has been changed.

A.37   Delete POP3 Accounts

- **path to WSDL file:**

  - [domain] /wsdl/deletePOP3Accounts.wsdl

- **input parameter:**

  - sessionID (type `string`): The session ID of the session used to log in with the account from which POP3 accounts are to be deleted.
  - pop3AccountToDeleteID (type `int`): The ID of a POP3 account to be deleted. This field must occur at least once and can occur any number of times to specify all IDs of POP3 accounts to delete (corresponding to an array of type `int` with at least 1 element).

- **return value:**

  - deleted (type `boolean`): Indicates that the operation was performed without error.

A.38   Retrieve mails for a POP3 account

- **path to WSDL file:**

  - [domain] /wsdl/pollPOP3Account.wsdl

- **input parameter:**

  - sessionID (type `string`): The session ID of the session used to log in to the account from whose POP3 account mail is to be retrieved.
  - pop3AccountToPollID (type `int`): The ID of the POP3 account to retrieve emails from.

- **return value:**

  - incomingMailsNotification (Type `incomingMailsNotification`): A notification of new incoming mails. This field can occur not once, once or twice, depending on in which folders (`Inbox` or `Spam` folder) new mails have arrived (corresponds to an array of the type `incomingMailsNotification` with at most 2 elements). The type `incomingMailsNotification` consists of the following fields:

∗ `folderID` (type `string`): The ID of the folder that received new mail.

∗ `usedStorageInByte` (type `long`): The amount of used storage of the account after the receipt of the new mail.

∗ `incomingMail` (type `mail`): A received mail. This field occurs at least once and can otherwise occur as often as required in order to transmit all incoming mail (corresponds to an array of the type `mail` with at least 1 element).

∗ `pop3ServerErrorMessages` (type `string`): Possible error messages that occurred during retrieval.

The type `mail` consists of the following fields:

∗ `mailID` (type `string`): The ID of the mail in the folder with the ID `folderID`. This ID is used to reference the mail in this folder.

∗ `senderAddress` (type `string`): The sender address of the mail.

∗ `sender` (type `string`): The sender of the mail. The field is only transmitted if it is not empty.

∗ `recipientAddress` (type `string`): A recipient address of the mail. If there are several recipients, the first recipient address is specified here.

∗ `recipient` (type `string`): A recipient of the mail. If there are several recipients, the first recipient is specified here. The field is only transmitted if it is not empty.

∗ `to` (type `string`): The full "To" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

∗ `cc` (type `string`): The complete "CC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

∗ `bcc` (type `string`): The complete "BCC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

∗ `replyTo` (type `string`): The "Reply-To" header field. The field is only transmitted if it exists in the header of the mail.

∗ `dispoTo` (type `string`): The "Disposition-Notification-To" header field. The field is only transmitted if it exists in the header of the mail.

∗ `subject` (type `string`): The subject of the mail. The field is only transmitted if it exists in the header of the mail and is not empty.

∗ `dateOfMail` (type `dateTime`): The time and date of the mail.

∗ `mailSizeInByte` (type `int`): The size of the mail in bytes.

∗ `wasReplied` (type `boolean`): Indicates whether the mail has been answered.

∗ `wasForwarded` (type `boolean`): Indicates whether the mail has been forwarded.

∗ `wasRead` (type `boolean`): Indicates whether the mail has been read.

∗ `attachments` (type `attachment`): The attachments of this mail. This field can occur any number of times to transfer all attachments of the mail (corresponds to an array of type `attachment`). The type `attachment` consists of the following fields:

· `nameOrFilename` (type `string`): The name or filename of the attachment.

· `fileSizeInByte` (type `int`): The size of the attachment in bytes.

· `contentID` (type `string`): The content ID of the attachment if specified.

· `contentLocation` (type `string`): The content location of the attachment if specified.

A.39   Retrieve mail for all POP3 accounts

- **path to WSDL file:**

  – `[domain] /wsdl/pollAllPOP3Accounts.wsdl`

- **input parameter:**

  - `sessionID` (type `string`): The session ID of the session used to log in with the account from whose POP3 accounts mail is to be retrieved.

- **return value:**

  - `incomingMailsNotification` (Type `incomingMailsNotification`): A notification of new incoming mails. This field can occur not once, once or twice, depending on in which folders (`Inbox` or `Spam` folder) new mails have arrived (corresponds to an array of the type `incomingMailsNotification` with at most 2 elements). The type `incomingMailsNotification` consists of the following fields:

    * `folderID` (type `string`): The ID of the folder that received new mail.
    * `usedStorageInByte` (type `long`): The amount of used storage of the account after the receipt of the new mail.
    * `incomingMail` (type `mail`): A received mail. This field occurs at least once and can otherwise occur as often as required in order to transmit all incoming mail (corresponds to an array of the type `mail` with at least 1 element).
    * `pop3ServerErrorMessages` (type `string`): Possible error messages that occurred during retrieval.

    The type `mail` consists of the following fields:

    * `mailID` (type `string`): The ID of the mail in the folder with the ID `folderID`. This ID is used to reference the mail in this folder.
    * `senderAddress` (type `string`): The sender address of the mail.
    * `sender` (type `string`): The sender of the mail. The field is only transmitted if it is not empty.
    * `recipientAddress` (type `string`): A recipient address of the mail. If there are several recipients, the first recipient address is specified here.
    * `recipient` (type `string`): A recipient of the mail. If there are several recipients, the first recipient is specified here. The field is only transmitted if it is not empty.
    * `to` (type `string`): The full "To" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
    * `cc` (type `string`): The complete "CC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
    * `bcc` (type `string`): The complete "BCC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
    * `replyTo` (type `string`): The "Reply-To" header field. The field is only transmitted if it exists in the header of the mail.
    * `dispoTo` (type `string`): The "Disposition-Notification-To" header field. The field is only transmitted if it exists in the header of the mail.
    * `subject` (type `string`): The subject of the mail. The field is only transmitted if it exists in the header of the mail and is not empty.
    * `dateOfMail` (type `dateTime`): The time and date of the mail.
    * `mailSizeInByte` (type `int`): The size of the mail in bytes.
    * `wasReplied` (type `boolean`): Indicates whether the mail has been answered.
    * `wasForwarded` (type `boolean`): Indicates whether the mail has been forwarded.
    * `wasRead` (type `boolean`): Indicates whether the mail has been read.
    * `attachments` (type `attachment`): The attachments of this mail. This field can occur any number of times to transfer all attachments of the mail (corresponds to an array of type `attachment`). The type `attachment` consists of the following fields:

· `nameOrFilename` (type `string`): The name or filename of the attachment.

· `fileSizeInByte` (type `int`): The size of the attachment in bytes.

· `contentID` (type `string`): The content ID of the attachment if specified.

· `contentLocation` (type `string`): The content location of the attachment if specified.

A.40   Retrieve existing subfolders of a folder

- **path to WSDL file:**

  – [domain] `/wsdl/getFolders.wsdl`

- **input parameter:**

  – `sessionID` (type `string`): The session ID of the session used to log in with the account for which existing subfolders of a folder are to be retrieved.

  – `parentFolderID` (type `string`): The ID of the parent folder. If `parentFolderID` is blank, it indicates that you want to get the folders at the top level.

- **return value:**

  – `folder` (type `folder`): An existing folder of this account. This field can occur as often as required to transfer all existing folders (corresponds to an array of the type `folder`). The type `folder` consists of the following fields:

    * `folderID` (type `string`): The ID of the folder. This ID is used to reference the folder.

    * `folderName` (type `string`): The name of the folder.

    * `numberOfMails` (type `int`): The number of mails in this folder.

    * `numberOfUnreadMails` (type `int`): The number of unread mail in this folder.

    * `numberOfBytes` (type `long`): The number of bytes of this folder.

    * `hasSubFolders` (type `boolean`): Indicates whether the folder has subfolders.

A.41   Create a Folder

- **path to WSDL file:**

  – [domain] `/wsdl/createFolder.wsdl`

- **input parameter:**

  – `sessionID` (type `string`): The session ID of the session used to log in with the account for which a folder is to be created.

  – `parentFolderID` (type `string`): The ID of the folder in which the new folder is to be created. If `parentFolderID` is empty, it indicates that the folder is to be created at the top level.

  – `newFolderName` (type `string`): The name of the new folder to create.

- **return value:**

  – `created` (type `boolean`): Indicates whether the folder was created.

  – `newFolderID` (type `string`): The ID of the newly created folder.

Note: If there is already a folder named `newFolderName`, an error message will be returned.

A.42   Rename a folder

- **path to WSDL file:**

  – [domain] /wsdl/renameFolder.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account of which a folder is to be renamed.

  – folderToRenameID (type string): The ID of the folder to be renamed.

  – newFolderName (type string): The new folder name.

- **return value:**

  – renamed (type boolean): Indicates that the operation was performed without error.

A.43   Move a folder

- **path to WSDL file:**

  – [domain] /wsdl/moveFolder.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account of which a folder is to be moved.

  – folderToMoveID (type string): The ID of the folder to be moved.

  – newParentFolderID (type string): The ID of the new parent folder of the folder to be moved. If the folder is to be moved to the top level, newParentFolderID must be empty.

- **return value:**

  – moved (type boolean): Indicates that the operation was performed without error.

A.44   Delete a folder

- **path to WSDL file:**

  – [domain] /wsdl/deleteFolder.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account of which a folder is to be deleted.

  – folderToDeleteID (type string): The ID of the folder to be deleted.

- **return value:**

  – deleted (type boolean): Indicates that the operation was performed without error.

Note: The folders Inbox, Sent messages, Deleted objects and Spam can not be deleted. If a folder to be deleted is not in the Deleted objects folder, it will be moved to that folder first. If a folder to be deleted is in this or one of its subfolders, it will be deleted.

A.45   Empty a folder

- **path to WSDL file:**

  - [domain] /wsdl/clearFolder.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account of which a folder is to be emptied.

  - folderToClearID (type string): The ID of the folder to be emptied.

- **return value:**

  - cleared (type boolean): Indicates that the operation was performed without error.

Note: This action can not be undone. The call can be called for all folders. However, the Web-ServiceClient only offers it for the Deleted objects and Spam folders so that folders containing relevant emails are not accidentally emptied.

A.46   Register for notifications about changes to the account

- **path to WSDL file:**

  - [domain] /wsdl/notify.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in to the account of which is to be notified of changes to the account.

  - runningNo (type int): The running number of the last notification received by the client. This number is given in the HTTP header from the server. If specified, the server transfers all cached notification transferred to the session with session ID sessionID after the specified running number.

- **return value:**

  - incomingMailsNotification (Type incomingMailsNotification): A notification of the receipt of new mails. This field can be null. If it is not null, it will occur either once or twice to notify you of new mail in the Inbox and/or Spam folder.
    The type incomingMailsNotification consists of the following fields:

    * folderID (type string): The ID of the folder that received new mail.
    * usedStorageInByte (type long): The amount of used storage of the account after the receipt of the new mail.
    * incomingMail (type mail): A received mail. This field occurs at least once and can otherwise occur as often as required in order to transmit all incoming e-mails (corresponds to an array of the type mail with at least 1 element).

    The type mail consists of the following fields:

    * mailID (type string): The ID of the mail in the folder with the ID folderID. This ID is used to reference the mail in this folder.
    * senderAddress (type string): The sender address of the mail.
    * sender (type string): The sender of the mail. The field is only transmitted if it is not empty.
    * recipientAddress (type string): A recipient address of the mail. If there are several recipients, the first recipient address is specified here.

* recipient (type string): A recipient of the mail. If there are several recipients, the first recipient is specified here. The field is only transmitted if it is not empty.
* to (type string): The full "To" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
* cc (type string): The full "CC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
* bcc (type string): The complete "BCC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.
* replyTo (type string): The "Reply-To" header field. The field is only transmitted if it exists in the header of the mail.
* dispoTo (type string): The "Disposition-Notification-To" header field. The field is only transmitted if it exists in the header of the mail.
* subject (type string): The subject of the mail. The field is only transmitted if it exists in the header of the mail and is not empty.
* dateOfMail (type dateTime): The time and date of the mail.
* mailSizeInByte (type int): The size of the mail in bytes.
* wasReplied (type boolean): Indicates whether the mail has been answered.
* wasForwarded (type boolean): Indicates whether the mail has been forwarded.
* wasRead (type boolean): Indicates whether the mail has been read.
* attachments (type attachment): The attachments of this mail. This field can occur any number of times to transfer all attachments of the mail (corresponds to an array of type attachment). The type attachment consists of the following fields:
  · nameOrFilename (type string): The name or filename of the attachment.
  · fileSizeInByte (type int): The size of the attachment in bytes.
  · contentID (type string): The content ID of the attachment if specified.
  · contentLocation (type string): The content location of the attachment if specified.

− mailChangedNotification (type mailChangedNotification): A notification that an email has been modified in the Drafts folder. This field can be null. The type mailChangedNotification consists of the following fields:

* usedStorageInByte (type long): The amount of used storage of the account after the mail was changed.
* changedInFolderID (type string): The ID of the Drafts folder in which an email was changed.
* newMailID (type string): The new ID of the modified mail. Can match the previous ID. If not, the mail can only be referenced by the new ID.
* changedMail (type mail): The changed mail.

− mailsAddedNotification: A notification that new mail has been added to a folder. This field can be null. The type mailsAddedNotification consists of the following fields:

* addedToFolderID (type string): The ID of the folder to which new mails have been added.
* usedStorageInByte (type long): The amount of used storage of the account after adding new mail.
* addedMail (type mail): An added mail. This field occurs at least once and can otherwise occur as often as required in order to transfer all added mails (corresponds to an array of the type mail with at least 1 element).

- mailsMovedNotification: A notification that mails have been moved to a different folder. This field can be null. The type mailsMovedNotification consists of the following fields:

    * usedStorageInByte (type long): The amount of used storage of the account, after the mails were moved, in bytes.
    * fromFolderID (type string): The ID of the folder from which the mails were moved.
    * toFolderID (type string): The ID of the folder to which the mails have been moved.
    * numberOfUnreadMailsFromFolder (type int): The number of unread messages after moving in the folder from which the mails were moved.
    * numberOfUnreadMailsToFolder (type int): The number of unread messages after moving in the folder to which the mails were moved.
    * movedMailID (type string): The ID of a moved mail. This field occurs at least once and can otherwise occur as often as required in order to transfer the IDs of all moved mails (corresponds to an array of the type mail with at least 1 element).

- mailsDeletedNotification (type mailsDeletedNotification): A notification that emails have been deleted from a folder. This field can be null. The type mailsDeletedNotification consists of the following fields:

    * folderID (type string): The ID of the folder from which the mails were deleted.
    * numberOfUnreadMails (type int): The number of unread mails after deletion in the folder from which the mails were deleted.
    * usedStorageInByte (type long): The amount of used storage of the account after deleting the mails in bytes.
    * deletedMailID (type string): The ID of a deleted mail. This field occurs at least once and can otherwise occur as often as required in order to transmit the IDs of all deleted mails (corresponds to an array of the type mail with at least 1 element).

- mailWasRepliedNotification (type mailWasRepliedNotification): A notification that an email has been replied. This field can be null. The type mailWasRepliedNotification consists of the following fields:

    * folderID (type string): The ID of the folder in which the replied mail is stored.
    * repliedMailID (type string): The ID of the replied mail.

- mailWasForwardedNotification (type mailWasForwardedNotification): A notification that a message has been forwarded. This field can be null. The type mailWasForwardedNotification consists of the following fields:

    * folderID (type string): The ID of the folder where the forwarded mail is stored.
    * forwardedMailID (type string): The ID of the forwarded mail.

- mailWasReadStateChangedNotification (type mailWasReadStateChangedNotification): A notification that the status "was read" of mails has been changed. This field can be null. The type mailWasReadStateChangedNotification consists of the following fields:

    * folderID (type string): The ID of the folder where the mail is stored, whose "was read" status has been changed.
    * numberOfUnreadMails (type int): The number of unread mail in the folder after which the status of the mail has been changed.
    * changedMailID (type string): The ID of a mails whose "was read" status has been changed. This field occurs at least once and can otherwise occur as often as required in order to transfer the IDs of all changed mails (corresponds to an array of the type string with at least 1 element).

* wasRead (type boolean): Whether or not the mail has been given the status "was read".

– newFolderAddedNotification (type newFolderAddedNotification): A notification that a new mail folder has been created. This field can be null. The type newFolderAddedNotification consists of the following fields:

* newFolderID (type string): The ID of the new folder.
* newFolderName (type string): The name of the new folder.
* parentFolderID (type string): The ID of the parent folder of the new folder.

– folderMovedNotification (type folderMovedNotification): A notification that a mail folder has been moved. This field can be null. The type folderMovedNotification consists of the following fields:

* movedFolderID (type string): The ID of the moved folder.
* toParentFolderID (type string): The ID of the new parent folder.

– folderRenamedNotification (type folderRenamedNotification): A notification that a mail folder has been renamed. This field can be null. The type folderRenamedNotification consists of the following fields:

* renamedFolderID (type string): The ID of the renamed folder.
* newFolderName (type string): The new name of the folder.

– folderDeletedNotification (type folderDeletedNotification): A notification that a mail folder has been deleted. This field can be null. The type folderDeletedNotification consists of the following fields:

* deletedFolderID (type string): The ID of the deleted folder.
* usedStorageInByte (type long): The amount of used storage of the account after the folder has been deleted.

– folderClearedNotification (type folderClearedNotification): A notification that a mail folder has been emptied. This field can be null. The type folderClearedNotification consists of the following fields:

* clearedFolderID (type string): The ID of the emptied folder.
* usedStorageInByte (type long): The amount of used storage of the account after the folder was emptied, in bytes.

– addressDataAddedNotification (type addressDataAddedNotification): A notification that an address record has been added to the address book. This field can be null. The type addressDataAddedNotification consists of the following fields:

* addedAddressData (type addressData): The newly added address record. The type addressData consists of the following fields:
  · guid (type string): The unique ID of the address record.
  · isPublic (type boolean): Indicates whether the address record is marked as "public".
  · mailAddress (type string): The mail address of the contact.
  · imageFileContentBase64Encoded (type string): If an image has been added to this contact, the contents of the image file are Base64 encoded. If there is no image, an empty string is returned.
  · firstName (type string): The first name of the person of this contact.
  · lastName (type string): The last name of the person of this contact.
  · privateStreet (type string): The street of the private address of the person of this contact.

- · `privateZipCode` (type `string`): The postal code of the private address of the person of this contact.
- · `privateCity` (type `string`): The location of the private address of the person in contact.
- · `privateState` (type `string`): The state of the private address of the person in contact.
- · `privateCountry` (type `string`): The country of the private address of the person of this contact.
- · `privatePhoneNumber` (type `string`): The private phone number of the person of this contact.
- · `privateFax` (type `string`): The private fax number of the person of this contact.
- · `privateWebSite` (type `string`): The private web site of the person of this contact.
- · `birthday` (type `dateTime`): The birthday of the person of this contact.
- · `mobilePhoneNumber` (type `string`): The mobile phone number of the person of this contact.
- · `businessStreet` (type `string`): The street of the business address of the person of this contact.
- · `businessZipCode` (type `string`): The postal code of the business address of the person of this contact.
- · `businessCity` (type `string`): The location of the business address of the person of this contact.
- · `businessState` (type `string`): The state of the business address of the person of this contact.
- · `businessCountry` (type `string`): The country of the business address of the person of this contact.
- · `businessPhoneNumber` (type `string`): The business phone number of the person of this contact.
- · `businessFax` (type `string`): The business fax number of the person of this contact.
- · `businessWebSite` (type `string`): The business website of the person of this contact.
- · `role` (type `string`): The position in the company of the person of this contact.
- · `companyName` (type `string`): The name of the company of the person of this contact.
- · `miscellaneous` (type `string`): Miscellaneous about this contact.

— `addressDataChangedNotification` (type `addressDataChangedNotification`): A notification that an address record in the address book has been changed. This field can be `null`. The type `addressDataChangedNotification` consists of the following fields:

  ∗ `changedAddressData` (type `addressData`): The changed address record.

— `addressDataPublicStateChangedNotification`
(type `addressDataPublicStateChangedNotification`): A notification that the status "public" of address records has changed. This field can be `null`. The type `addressDataPublicStateChangedNotification` consists of the following fields:

  ∗ `changedGUID` (type `string`): The unique ID of an address record whose "public" status has been changed. This field occurs at least once and can otherwise occur as often as required in order to transfer the IDs of all changed address data records (corresponds to an array of the type `string` with at least 1 element).

* isPublic (type `boolean`): Whether the address records have been given the status "public" or not.

— addressDataDeletedNotification (type `addressDataDeletedNotification`): A notification that address records have been deleted from the address book. This field can be `null`. The type `addressDataDeletedNotification` consists of the following fields:

* deletedGUID (type `string`): The unique ID of a deleted address record. This field occurs at least once and can otherwise occur as often as required in order to transfer the unique IDs of all deleted address data records (corresponds to an array of the type `string` with at least of 1 element).

— addressGroupAddedNotification (type `addressGroupAddedNotification`): A notification that an address group has been added to the address book. This field can be `null`. The type `addressGroupAddedNotification` consists of the following fields:

* addressGroupName (type `string`): The name of the added address group.
* guid (type `string`): The unique ID of an address record of the address group to be added. This field can occur as often as required in order to transfer all unique IDs of the address records contained by the added address group (corresponds to an array of type `string`).

— addressGroupChangedNotification (type `addressGroupChangedNotification`): A notification that an address group has changed. This field can be `null`. The type `addressGroupChangedNotification` consists of the following fields:

* addressGroupName (type `string`): The name of the changed address group.
* guid (type `string`): The unique ID of an address record of the changed address group. This field can occur as often as required in order to transfer all unique IDs of the address records contained by the changed address group (corresponds to an array of type `string`).

— addressGroupRenamedNotification (type `addressGroupRenamedNotification`): A notification that an address data group has been renamed. This field can be `null`. The type `addressGroupRenamedNotification` consists of the following fields:

* addressGroupName (type `string`): The former name of the renamed address group.
* newGroupName (type `string`): The new name of the address group.

— addressGroupsDeletedNotification (type `addressGroupsDeletedNotification`): A notification that an address group has been deleted from the address book. This field can be `null`. The type `addressGroupsDeletedNotification` consists of the following fields:

* addressGroupName (type `string`): The name of a deleted address group. This field occurs at least once and otherwise occur as often as required in order to transfer all deleted address groups (corresponds to an array of the type `string` with at least 1 element).

— calendarEntryAddedNotification (Type `calendarEntryAddedNotification`): A notification that an event has been added to the calendar. This field can be `null`. The type `calendarEntryAddedNotification` consists of the following fields:

* calendarEntry (Type `calendarEntry`): The added appointment. The type `calendarEntry` consists of the following fields:
  · isPublic (type `boolean`): Indicates whether the entry was marked as "public".
  · guid (type `string`): The GUID of the calendar entry.
  · summary (type `string`): Brief summary of the event denoted by this calendar entry.
  · startTime (type `dateTime`): The time at which the event denoted by this calendar entry begins.

· `guid` (type `string`): The time at which the event denoted by this calendar entry ends.

· `reminderInMinutesBeforeStart` (type `int`): The number of minutes before `startTime` to remind of the beginning of the event denoted by this calendar entry. The value of -1 indicates that you do not want to be reminded.

· `reminderRecipient` (type `string`): An e-mail address to which a reminder of this appointment should be sent to at the time specified by `reminderInMinutesBeforeStart`. This field can occur any number of times to transfer all recipients of the reminder mail (corresponds to an array of type `string`).

· `location` (type `string`): The location where the event denoted by this calendar entry is to take place.

· `resources` (type `string`): The resources that the event denoted by this calendar entry occupies.

· `description` (type `string`): Detailed description of the event denoted by this calendar entry.

· `rrule` (type `string`): The recurrence rule of the event denoted by this calendar entry in iCalendar format.

− `calendarEntryChangedNotification` (type `calendarEntryChangedNotification`): A notification that a calendar entry has been changed. This field can be `null`. The type `calendarEntryChangedNotification` consists of the following fields:

∗ `calendarEntry` (type `calendarEntry`): The modified calendar entry.

− `calendarEntriesPublicStateChangedNotification` (type `calendarEntriesPublicStateChangedNotification`): A notification that the "public" status of calendar entries has changed. This field can be `null`. The type `calendarEntriesPublicStateChangedNotification` consists of the following fields:

∗ `changedCalendarEntryGUID` (type `string`): The GUID of a calendar entry whose "public" status has changed. This field occurs at least once and can otherwise occur as often as required in order to transfer the GUIDs of all changed calendar entries (corresponds to an array of type `string` with at least 1 element).

∗ `isPublic` (type `boolean`): Whether the calendar entries have been given the status "public" or not.

− `calendarEntriesDeletedNotification` (type `calendarEntriesDeletedNotification`): A notification that entries have been deleted from the calendar. This field can be `null`. The type `calendarEntriesDeletedNotification` consists of the following fields:

∗ `deletedCalendarEntryGUID` (type `string`): The GUID of a deleted calendar entry. This field occurs at least once and can otherwise occur as often as required in order to transfer the GUIDs of all deleted calendar entries (corresponds to an array of type `string` with at least 1 element).

− `filesAddedNotification` (type `filesAddedNotification`): A notification that files have been uploaded to the file manager. This field can be `null`. The type `filesAddedNotification` consists of the following fields:

∗ `file` (type `file`): An uploaded file. The type `file` consists of the following fields:
  · `isPublic` (type `boolean`): Indicates whether the file has been marked as "public".
  · `guid` (type `string`): The GUID of the file.
  · `fileName` (type `string`): The name of the file.
  · `uploadDateTime` (type `dateTime`): The time when the file was uploaded.
  · `fileSizeInByte` (type `long`): The size of the file in bytes.
  · `description` (type `string`): Description of the file.

- − fileChangedNotification (type fileChangedNotification): A notification that a file has been modified. This field can be null.
  The type fileChangedNotification consists of the following fields:
    - ∗ file (type file): The modified file.
- − filesPublicStateChangedNotification (type filesPublicStateChangedNotification): A notification that the "public" status of files has changed. This field can be null. The type filesPublicStateChangedNotification consists of the following fields:
    - ∗ changedFileGUID (type string): The GUID of a file whose '"public" status has changed. This field occurs at least once and can otherwise occur as often as required in order to transfer the GUIDs of all changed files (corresponds to an array of the type string with at least 1 element).
    - ∗ isPublic (type boolean): Whether the files have been given the status "public" or not.
- − filesDeletedNotification (type filesDeletedNotification): A notification that files have been deleted from the file manager. This field can be null. The type filesDeletedNotification consists of the following fields:
    - ∗ deletedFileGUID (type string): The GUID of a deleted file. This field occurs at least once and can otherwise occur as often as required in order to transfer the GUIDs of all deleted files (corresponds to an array of the type string with at least 1 element).

Note: In order to be informed about the changes of other users in the account when multiple accounts are used at the same time, as well as to avoid regular queries about new incoming mails, this web service call is offered. It blocks until a notification. Even after a notification, the socket is not closed on the server side, but kept open for further notifications. It is recommended that the client continues to use this socket and does not issue a new "notify" call. If a new "notify" call is issued by a client, the previous socket is closed on the server side.

A.47   Retrieve mails in a folder

- **path to WSDL file:**
    - − [domain] /wsdl/getMails.wsdl
- **input parameter:**
    - − sessionID (type string): The session ID of the session used to log in with the account for which the mails of a folder are to be retrieved.
    - − folderID (type string): The ID of the folder whose mails are to be retrieved.
    - − newerThan (type dateTime): Specifies a timestamp. Only mails whose timestamps are newer than the specified timestamp will be returned.
- **return value:**
    - − usedStorageInByte (type long): Currently used storage by the account in bytes. To keep this information as accurate as possible, it will also be included in this call.
    - − numberOfMails (type int): The number of mails stored in the folder with the ID folderID.
    - − numberOfBytes (type long): The total number of bytes of all mails stored in the folder with the ID folderID.
    - − mail (type mail): A mail in the folder with the ID folderID. This field can occur any number of times to transfer all mails in the folder (corresponds to an array of the type mail).
      The type mail consists of the following fields:

* mailID (type string): The ID of the mail in the folder with the ID folderID. This ID is used to reference the mail in this folder.

* senderAddress (type string): The sender address of the mail.

* sender (type string): The sender of the mail. The field is only transmitted if it is not empty.

* recipientAddress (type string): A recipient address of the mail. If there are several recipients, the first recipient address is specified here.

* recipient (type string): A recipient of the mail. If there are several recipients, the first recipient is specified here. The field is only transmitted if it is not empty.

* to (type string): The full "To" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

* cc (type string): The complete "CC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

* bcc (type string): The complete "BCC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

* replyTo (type string): The "Reply-To" header field. The field is only transmitted if it exists in the header of the mail.

* dispoTo (type string): The "Disposition-Notification-To" header field. The field is only transmitted if it exists in the header of the mail.

* subject (type string): The subject of the mail. The field is only transmitted if it exists in the header of the mail and is not empty.

* dateOfMail (type dateTime): The time and date of the mail.

* mailSizeInByte (type int): The size of the mail in bytes.

* wasReplied (type boolean): Indicates whether the mail has been answered.

* wasForwarded (type boolean): Indicates whether the mail has been forwarded.

* wasRead (type boolean): Indicates whether the mail has been read.

* attachments (type attachment): The attachments of this mail. This field can occur any number of times to transfer all attachments of the mail (corresponds to an array of type attachment). The type attachment consists of the following fields:

  · nameOrFilename (type string): The name or filename of the attachment.

  · fileSizeInByte (type int): The size of the attachment in bytes.

  · contentID (type string): The content ID of the attachment if specified.

  · contentLocation (type string): The content location of the attachment if specified.

A.48   Search a folder for mails

- **path to WSDL file:**

  - [domain] /wsdl/searchMails.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account for which a folder is to be searched for mails.

  - folderID (type string): The ID of the folder to be searched for mails.

  - includeSubFolders (type boolean): Specifies whether the subfolders of the folder with the ID folderID are to be included in the search.

  - from (type string): The string to search in the sender field of the mail. Case-insensitive. If the field is left blank, it will not be used as a search criterion.

 – to (type string): The string to search in one of the recipient fields of the mail (To, CC, BCC). Case-insensitive. If the field is left blank, it will not be used as a search criterion.

 – subject (type string): The string to search in the subject field of the message. Case-insensitive. If the field is left blank, it will not be used as a search criterion.

 – messageBody (type string): The string to search in the body of the mail. Case-insensitive. If the field is left blank, it will not be used as a search criterion.

 – before (type boolean): If set to true, only those emails will match the search criteria whose date precedes the date beforeMailDate.

 – after (type boolean): If set to true, only those mails will match the search criteria whose date is after the date afterMailDate.

 – beforeMailDate (type dateTime): The date before which the date of the searched mail should be, if before was set to true.

 – afterMailDate (type dateTime): The date after which the date of the searched mail should be, if after was set to true.

 – mailHasAttachements (type boolean): If set to true, only those emails will match the search criteria that have an attachment.

 – fileName (type string): The string to search in the file names of files attached to the searched mail. Case-insensitive. If the field is left blank, it will not be used as a search criterion.

- **return value:**

 – foundMail (type mail): An email in the folder with the ID folderID (or one of its subfolders) matching the search criteria. This field can occur any number of times to transfer all mails found (corresponds to an array of the type mail).
   The type mail consists of the following fields:

   * mailID (type string): The ID of the mail in the folder with the ID parentFolderID. This ID is used to reference the mail in this folder.

   * parentFolderID (type string): The ID of the parent folder of the mail with the ID mailID.

   * parentFolderName (type string): The name of the parent folder of the mail with the id mailID.

   * senderAddress (type string): The sender address of the mail.

   * sender (type string): The sender of the mail. The field is only transmitted if it is not empty.

   * recipientAddress (type string): A recipient address of the mail. If there are several recipients, the first recipient address is specified here.

   * recipient (type string): A recipient of the mail. If there are several recipients, the first recipient is specified here. The field is only transmitted if it is not empty.

   * to (type string): The full "To" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

   * cc (type string): The full "CC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

   * bcc (type string): The complete "BCC" header field. The field is only transmitted if it exists in the header of the mail and is not empty.

   * replyTo (type string): The "Reply-To" header field. The field is only transmitted if it exists in the header of the mail.

   * dispoTo (type string): The "Disposition-Notification-To" header field. The field is only transmitted if it exists in the header of the mail.

* subject (type string): The subject of the mail. The field is only transmitted if it exists in the header of the mail and is not empty.
* dateOfMail (type dateTime): The time and date of the mail.
* mailSizeInByte (type int): The size of the mail in bytes.
* wasReplied (type boolean): Indicates whether the mail has been answered.
* wasForwarded (type boolean): Indicates whether the mail has been forwarded.
* wasRead (type boolean): Indicates whether the mail has been read.
* attachments (type attachment): The attachments of this mail. This field can occur any number of times to transfer all attachments of the mail (corresponds to an array of type attachment). The type attachment consists of the following fields:
  · nameOrFilename (type string): The name or filename of the attachment.
  · fileSizeInByte (type int): The size of the attachment in bytes.
  · contentID (type string): The content ID of the attachment if specified.
  · contentLocation (type string): The content location of the attachment if specified.

Note: If a search is to be performed on all existing folders, the field folderID in the request must be empty and the field includeSubFolders must be set to "true".

A.49   Delete mails

- **path to WSDL file:**
  - [domain] /wsdl/deleteMails.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which mails are to be deleted.
  - folderID (type string): The ID of the folder from which emails are to be deleted.
  - mailID (type string): The ID of a mail to be deleted. This field must occur at least once and may otherwise occur as often as required to transfer the IDs of all emails to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**
  - deleted (type boolen): Indicates that the operation was performed without error.
  - usedStorageInByte (type long): The storage used after the mails have been deleted.

A.50   Move mails

- **path to WSDL file:**
  - [domain] /wsdl/moveMails.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which mails are to be moved.
  - toSessionID (type string): The session ID of the session used to log in with the account into whose folder with id toFolderID mails are to be moved. Can be left away if mails are to be moved between folders within the same account.

– `fromFolderID` (type `string`): The ID of the folder from which the mails are to be moved.

– `toFolderID` (type `string`): The ID of the folder to which the mails are to be moved.

– `mailID` (type `string`): The ID of a mail to be moved. This field must occur at least once and may otherwise occur as often as required to transfer the IDs of all mails to be moved (corresponds to an array of the type `string` with at least 1 element).

– `addSenderAddressesToWhitelist` (type `boolean`): Specifies whether the sender addresses of moved emails should be whitelisted. This is especially useful if there are false positives in the `Spam` folder after a filtering process that you want to move to the `Inbox`.

- **return value:**

  – `moved` (type `boolean`): Indicates whether the specified mails have been moved.

Notes:

- Mails can only be moved between accounts of the same FB-MailService instance.

- Moving mails between two different accounts is much more expensive than moving mails within the same account.

A.51   Copy mails

- **path to WSDL file:**

  – `[domain] /wsdl/copyMails.wsdl`

- **input parameter:**

  – `sessionID` (type `string`): The session ID of the session used to log in with the account for mails are to be copied.

  – `toSessionID` (type `string`): The session ID of the session used to log in with the account into whose folder with id `toFolderID` mails are to be copied. Can be left away if mails are to be copied between folders within the same account.

  – `fromFolderID` (type `string`): The ID of the folder from which the mails are to be copied.

  – `toFolderID` (type `string`): The ID of the folder to which the mails are to be copied.

  – `mailID` (type `string`): The ID of a mail to be copied. This field must occur at least once and may otherwise occur as often as required to transfer the IDs of all mails to be copied (corresponds to an array of the type `string` with at least 1 element).

- **return value:**

  – `copied` (type `boolean`): Indicates whether the specified mails have been copied.

Notes:

- Mails can only be copied between accounts of the same FB-MailService instance.

- Copying mails between two different accounts is much more expensive than copying mails within the same account.

A.52   Upload mails to the file management

- **path to WSDL file:**

  – [domain] /wsdl/uploadMails.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which e-mails are to be uploaded.

  – folderID (type string): The ID of the folder from which the mail are to be uploaded.

  – mailID (type string): The ID of a mail to be uploaded. This field must occur at least once and may otherwise occur as often as required to transfer the IDs of all mails to be uploaded (corresponds to an array of the type string with at least 1 element).

- **return value:**

  – uploaded (type boolean): Indicates whether the specified mails have been uploaded to the file management.

A.53   Change if an email has been replied

- **path to WSDL file:**

  – [domain] /wsdl/setWasReplied.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which the status of a mail is to be set to "replied".

  – folderID (type string): The ID of the folder whose mail is to be set to the status "replied".

  – mailID (type string): The ID of the mail whose status is to be set to "replied".

- **return value:**

  – set (type boolean): Indicates if the status has been set.

A.54   Change if a mail has been forwarded

- **path to WSDL file:**

  – [domain] /wsdl/setWasForwarded.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session that is logged on with the account for which the status of a mail is to be set to "forwarded".

  – folderID (type string): The ID of the folder whose mail is to set to the status "forwarded".

  – mailID (type string): The ID of the mail whose status is to be set to "forwarded".

- **return value:**

  – set (type boolean): Indicates if the status has been set.

A.55   Change if a mail was read or not read

- **path to WSDL file:**

    - [domain] /wsdl/setWasReadState.wsdl

- **input parameter:**

    - sessionID (type string): The session ID of the session used to log in with the account for which the status of a mail is to be set to "read"/"unread".
    - folderID (type string): The ID of the folder whose mail is to be set to "read"/"unread" status.
    - wasRead (type boolen): Whether the status of the mail should be set to "read" or "unread".
    - mailID (type string): The ID of the mail whose status is to be set to "read"/"unread". This field must occur at least once and may otherwise occur as often as required to transfer the IDs of all mails whose status is to be set (corresponds to an array of the type string with at least 1 element).

- **return value:**

    - set (type boolean): Indicates if the status has been set.

A.56   Retrieve Auto Respond/Forward Settings

- **path to WSDL file:**

    - [domain] /wsdl/getAutoRespondOrRelaySettings.wsdl

- **input parameter:**

    - sessionID (type string): The session ID of the session used to log in with the account for which the auto respond/forward settings are to be retrieved.

- **return value:**

    - autoRespondOrRelayMails (type boolean): Indicates whether incoming mails that have not been classified as spam are to be automatically responded and/or forwarded.
    - autoRespondFromDateTime (Type dateTime): The time from which mails are to be automatically responded/forwarded.
    - autoRespondToDateTime (Type dateTime): The time when emails are no longer to be responded/forwarded automatically.
    - autoRespondMails (type boolean): Indicates whether incoming mails that have not been classified as spam are to be responded automatically.
    - autoRespondSubject (type string): The subject of the mail to be sent as an automatic reply message.
    - autoRespondContent (type string): The content of the mail to be sent as an automatic reply message.
    - autoRelayMails (type boolean): Indicates whether incoming mails that have not been classified as spam are to be forwarded automatically to the local account named autoRelayToAccountName.
    - autoRelayToAccountName (type string): The name of the local account to which incoming mails that are not considered spam are to be forwarded automatically.

A.57   Change Auto Respond/Forward Settings

- **path to WSDL file:**

  - [domain] /wsdl/setAutoRespondOrRelaySettings.wsdl

- **input parameter:**

  - `sessionID` (type `string`): The session ID of the session that is logged on with the account for which the auto respond/forward settings are to be changed.

  - `autoRespondOrRelayMails` (type `boolean`): Specifies that incoming mails that have not been classified as spam are to be responded and/or forwarded automatically.

  - `autoRespondFromDateTime` (Type `dateTime`): The time from which mails are to be responded/forwarded automatically.

  - `autoRespondToDateTime` (Type `dateTime`): The time when emails are no longer to be responded/forwarded automatically.

  - `autoRespondMails` (type `boolean`): Specifies that incoming mails that have not been classified as spam are to be responded automatically.

  - `autoRespondSubject` (type `string`): The subject of the mail to be sent as an automatic reply message.

  - `autoRespondContent` (type `string`): The content of the mail to be sent as an automatic reply message.

  - `autoRelayMails` (type `boolean`): Specifies that incoming mail that has not been classified as spam are to be forwarded automatically to the local account named `autoRelayToAccountName`.

  - `autoRelayToAccountName` (type `boolean`): The name of the local account to which incoming mails that were not considered spam are to be forwarded automatically.

- **return value:**

  - `set` (type `boolean`): Indicates whether the Auto Respond/Forward settings have been changed.

A.58   Retrieve Public Address Data

- **path to WSDL file:**

  - [domain] /wsdl/getPublicAddressData.wsdl

- **input parameter:**

  - `sessionID` (type `string`): The session ID of the session used to log in with the account used to retrieve public address data/groups of accounts having the same domain.
    `searchString` (type `string`): Optional keywords for a search. If `searchString` is specified and not empty, only address records with all keywords found in their fields will be returned.
    This is especially useful for mobile or low bandwidth clients.

- **return value:**

  - `accountWithPublicAddressData` (type `accountWithPublicAddressData`): An account of the same domain as the querying one that has at least one public address record. This field can occur any number of times to transfer all accounts that have public address data (equivalent to an array of type `accountWithPublicAddressData`).
    The type `accountWithPublicAddressData` consists of the following fields:

* accountName (type string): The name of an account of the same domain as the querying one, which has at least one public address record or one public address group.
* addressData (type addressData): A public address record of the account named accountName. This field can occur any number of times to transfer all the public address records of this account. The type addressData consists of the following fields:
  · guid (type string): The unique ID of the address record.
  · isPublic (type boolean): Indicates whether the address record is marked as "public".
  · mailAddress (type string): The mail address of the contact.
  · imageFileContentBase64Encoded (type string): If an image has been added to this contact, the contents of the image file are Base64 encoded. If there is no image, an empty string is returned.
  · firstName (type string): The first name of the person of this contact.
  · lastName (type string): The last name of the person of this contact.
  · privateStreet (type string): The street of the private address of the person of this contact.
  · privateZipCode (type string): The postal code of the private address of the person of this contact.
  · privateCity (type string): The location of the private address of the person of this contact.
  · privateState (type string): The state of the private address of the person of this contact.
  · privateCountry (type string): The country of the private address of the person of this contact.
  · privatePhoneNumber (type string): The private phone number of the person of this contact.
  · privateFax (type string): The private fax number of the person of this contact.
  · privateWebSite (type string): The private website of the person of contact.
  · birthday (type dateTime): The birthday of the person of this contact.
  · mobilePhoneNumber (type string): The mobile phone number of the person of this contact.
  · businessStreet (type string): The street of the business address of the person of this contact.
  · businessZipCode (type string): The postal code of the business address of the person of this contact.
  · businessCity (type string): The location of the business address of the person of this contact.
  · businessState (type string): The state of the business address of the person of this contact.
  · businessCountry (type string): The country of the business address of the person of this contact.
  · businessPhoneNumber (type string): The business phone number of the person of this contact.
  · businessFax (type string): The business fax number of the person of this contact.
  · businessWebSite (type string): The business website of the person of this contact.

· `role` (type `string`): The position in the company of the person of this contact
.

· `companyName` (type `string`): The name of the company of the person of this contact.

· `miscellaneous` (type `string`): Miscellaneous about this contact.

A.59   Retrieve an address book

- **path to WSDL file:**

  - `[domain] /wsdl/getAddressBook.wsdl`

- **input parameter:**

  - `sessionID` (type `string`): The session ID of the session used to log in with the account for which an address book is to be retrieved.

  - `accountName` (type `string`): The name of an account having the same domain as querying one whose public address information is to be retrieved.
    Note: If `accountName` is empty, the complete address book of the querying account will be returned. If `accountName` is not empty, the public address data and groups of the account with the name `accountName` will be returned, if this account belongs to the same domain as the querying account. `searchString` (type `string`): Optional keywords for a search. If `searchString` is specified and not empty, only address records with all keywords found in their fields will be returned.
    This is especially useful for mobile or low bandwidth clients.

- **return value:**

  - `addressData` (type `addressData`): A contact that is in the address book. This field can occur any number of times to transfer all contacts in the address book (corresponds to an array of type `addressData`). The type `addressData` consists of the following fields:

    * `guid` (type `string`): The unique ID of the address record.
    * `isPublic` (type `boolean`): Indicates whether the address record is marked as "public".
    * `mailAddress` (type `string`): The mail address of the contact.
    * `imageFileContentBase64Encoded` (type `string`): If an image has been added to this contact, the contents of the image file are Base64 encoded. If there is no image, an empty string is returned.
    * `firstName` (type `string`): The first name of the person of this contact.
    * `lastName` (type `string`): The last name of the person of this contact.
    * `privateStreet` (type `string`): The street of the private address of the person of this contact.
    * `privateZipCode` (type `string`): The postal code of the private address of the person of this contact.
    * `privateCity` (type `string`): The location of the private address of the person of this contact.
    * `privateState` (type `string`): The state of the private address of the person of this contact.
    * `privateCountry` (type `string`): The country of the private address of the person of this contact.
    * `privatePhoneNumber` (type `string`): The private phone number of the person of this contact.
    * `privateFax` (type `string`): The private fax number of the person of this contact.

* privateWebSite (type `string`): The private website of the person of this contact.
* birthday (type `dateTime`): The birthday of the person of this contact.
* mobilePhoneNumber (type `string`): The mobile phone number of the person of this contact.
* businessStreet (type `string`): The street of the business address of the person of this contact.
* businessZipCode (type `string`): The postal code of the business address of the person of this contact.
* businessCity (type `string`): The location of the business address of the person of this contact.
* businessState (type `string`): The state of the business address of the person of this contact.
* businessCountry (type `string`): The country of the business address of the person of this contact.
* businessPhoneNumber (type `string`): The business phone number of the person of this contact.
* businessFax (type `string`): The business fax number of the person of this contact.
* businessWebSite (type `string`): The business website of the person of this contact.
* role (type `string`): The position in the company of the person of this contact.
* companyName (type `string`): The name of the company of the person of this contact.
* miscellaneous (type `string`): Miscellaneous about this contact.

  − addressGroup (type `addressGroup`): An address group that appears in the address book. This field can occur any number of times to transfer all address groups in the address book (corresponds to an array of type `addressGroup`). The type `addressGroup` consists of the following fields:

    * isPublic (type `boolean`): Indicates whether the address group has been marked as "public".
    * groupName (type `string`): The name of the address group.
    * mailAddress (type `string`): An e-mail address of the address group. This field can occur as often as required in order to transfer all mail addresses of the address group (corresponds to an array of type `string`)

A.60   Add address data to an address book

* **path to WSDL file:**

  − `[domain] /wsdl/addAddressData.wsdl`

* **input parameter:**

  − sessionID (type `string`): The session ID of the session used to log in with the account for which address data is to be added to the address book.
  − addressData: The address record to add to the address book. The type `addressData` consists of the following fields:

    * isPublic (type `boolean`): Indicates whether the address record is marked as "public".
    * mailAddress (type `string`): The mail address of the contact.
    * imageFileContentBase64Encoded (type `string`): If an image has been added to this contact, the contents of the image file are Base64 encoded. If there is no image, an empty string is returned.

* firstName (type string): The first name of the person of this contact.
* lastName (type string): The last name of the person of this contact.
* privateStreet (type string): The street of the private address of the person of this contact.
* privateZipCode (type string): The postal code of the private address of the person of this contact.
* privateCity (type string): The location of the private address of the person of this contact.
* privateState (type string): The state of the private address of the person of this contact.
* privateCountry (type string): The country of the private address of the person of this contact.
* privatePhoneNumber (type string): The private phone number of the person of this contact.
* privateFax (type string): The private fax number of the person of this contact.
* privateWebSite (type string): The private website of the person of this contact.
* birthday (type dateTime): The birthday of the person of this contact.
* mobilePhoneNumber (type string): The mobile phone number of the person of this contact.
* businessStreet (type string): The street of the business address of the person of this contact.
* businessZipCode (type string): The postal code of the business address of the person of this contact.
* businessCity (type string): The location of the business address of the person of this contact.
* businessState (type string): The state of the business address of the person of this contact.
* businessCountry (type string): The country of the business address of the person of this contact.
* businessPhoneNumber (type string): The business phone number of the person of this contact.
* businessFax (type string): The business fax number of the person of this contact.
* businessWebSite (type string): The business website of the person of this contact.
* role (type string): The position in the company of the person of this contact .
* companyName (type string): The name of the business of the person of this contact.
* miscellaneous (type string): Miscellaneous about this contact.

* **return value:**

  - added (type boolean): Indicates whether the address record has been added to the specified address book.
  - guid (type string): The unique ID of the newly added address record.

Note: The contact's e-mail address is always saved in lowercase letters, so mail addresses specified when contacts are to be changed or deleted must always be in lowercase letters.

A.61   Change existing address data in an address book

* **path to WSDL file:**

  - [domain] /wsdl/setAddressData.wsdl

- **input parameter:**
  - sessionID (type `string`): The session ID of the session used to log in with the account for which address data in the address book is to be changed.
  - addressData: The address record to be changed in the address book. The type addressData consists of the following fields:
    * guid (type `string`): The unique ID of the address record to modify.
    * isPublic (type `boolean`): Indicates whether the address record is marked as "public".
    * mailAddress (type `string`): The mail address of the contact.
    * imageFileContentBase64Encoded (type `string`): If an image has been added to this contact, the contents of the image file are Base64 encoded. If there is no image, an empty string is returned.
    * firstName (type `string`): The first name of the person of this contact.
    * lastName (type `string`): The last name of the person of this contact.
    * privateStreet (type `string`): The street of the private address of the person of this contact.
    * privateZipCode (type `string`): The postal code of the private address of the person of this contact.
    * privateCity (type `string`): The location of the private address of the person of this contact.
    * privateState (type `string`): The state of the private address of the person of this contact.
    * privateCountry (type `string`): The country of the private address of the person of this contact.
    * privatePhoneNumber (type `string`): The private phone number of the person of this contact.
    * privateFax (type `string`): The private fax number of the person of this contact.
    * privateWebSite (type `string`): The private website of the person of this contact.
    * birthday (type `dateTime`): The birthday of the person of this contact.
    * mobilePhoneNumber (type `string`): The mobile phone number of the person of this contact.
    * businessStreet (type `string`): The street of the business address of the person of this contact.
    * businessZipCode (type `string`): The postal code of the business address of the person of this contact.
    * businessCity (type `string`): The location of the business address of the person of contact.
    * businessState (type `string`): The state of the business address of the person of this contact.
    * businessCountry (type `string`): The country of the business address of the person of this contact.
    * businessPhoneNumber (type `string`): The business phone number of the person of this contact.
    * businessFax (type `string`): The business fax number of the person of this contact.
    * businessWebSite (type `string`): The business website of the person of this contact.
    * role (type `string`): The position in the company of the person of this contact.
    * companyName (type `string`): The name of the company of the person of this contact.

∗ miscellaneous (type string): Miscellaneous about this contact.

- **return value:**

  – set (type boolean): Indicates whether the address record was found and changed in the address book.

A.62 Change the "public" property of address data

- **path to WSDL file:**

  – [domain] /wsdl/setPublicStateOfAddressData.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for whose address data the property "public" is to be set.

  – isPublic (type boolean): Specifies what the "public" property should be set to.

  – guidToSet (type string): Specifies an address record based on its unique ID, whose "public" property is to be set to isPublic. This field must occur at least once and may otherwise occur as often as required in order to transfer all address data whose "public" property is to be set to isPublic (corresponds to an array of type string with at least 1 element).

- **return value:**

  – set (type boolean): Indicates whether the property "public" is set to isPublic for all specified address data found in the address book.

A.63 Delete existing address data from an address book

- **path to WSDL file:**

  – [domain] /wsdl/deleteAddressData.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which address data is to be deleted from the address book.

  – guidToDelete (type string): The unique ID of an address record to be deleted. This field must occur at least once and can occur any number of times to specify the IDs of all address records to be deleted (equivalent to an array of type string with at least 1 element).

- **return value:**

  – deleted (type boolean): Indicates that after the call, none of the specified address records are left in the address book.

A.64 Add an address group to an address book

- **path to WSDL file:**

  – [domain] /wsdl/addAddressGroup.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which an address group is to be added to the address book.

– groupNameToAdd (type string): The name of the address group to add.

– guid (type string): The ID of an address record of the address group to be added. This field can occur as often as required in order to transfer all IDs of the address data records of the address group to be added (corresponds to an array of the type string).

- **return value:**

  – added (type boolean): Indicates whether the group has been added to the address book.

A.65 Rename an address group

- **path to WSDL file:**

  – [domain] /wsdl/renameAddressGroup.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which an address group is to be renamed.

  – groupNameToRename (type string): The name of the address group to rename.

  – newGroupName (type string): The new name of the address group.

- **return value:**

  – renamed (type boolean): Indicates whether the group was found and renamed in the address book.

A.66 Change an address group

- **path to WSDL file:**

  – [domain] /wsdl/setAddressGroup.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which an address group is to be changed.

  – groupNameToSet (type string): The name of the address group to change.

  – guid (type string): The ID of an address record of the address group to be changed. This field can occur any number of times to transfer all IDs of the address data records of the address group to be changed (corresponds to an array of type string).

- **return value:**

  – set (type boolean): Indicates whether the group was found and changed in the address book.

A.67 Delete address groups from an address book

- **path to WSDL file:**

  – [domain] /wsdl/deleteAddressGroup.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which an address group is to be deleted from the address book.

– groupNameToDelete (type string): The name of an address group to be deleted. This field must occur at least once and can occur any number of times to specify all address groups to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**

– deleted (type boolean): Indicates that after the call, none of the specified address groups are left in the address book.

A.68   Retrieve existing templates

- **path to WSDL file:**

– [domain] /wsdl/getTemplates.wsdl

- **input parameter:**

– sessionID (type string): The session ID of the session used to log in with the account for which existing templates are to be retrieved.

- **return value:**

– template (type template): An existing template. This field can occur any number of times to transfer all existing templates of the account (corresponds to an array of type template). The type template consists of the following fields:

* templateName (type string): The name of the template.
* isHTML (type boolean): Whether the template is unformatted text or HTML formatted text.
* caretPosition (type int): The position of the cursor.
* fullQuote (type boolean): Whether emails created using this template contain the text of the answered or forwarded email as a full quote.
* contentBase64Encoded (type string): The content of the template as a Base64 encoded string.

A.69   Add a template

- **path to WSDL file:**

– [domain] /wsdl/addTemplate.wsdl

- **input parameter:**

– sessionID (type string): The session ID of the session used to log in with the account for which a template is to be added.
– templateNameToAdd (type string): The name of the template to add.
– isHTML (type boolean): Whether the template is unformatted text or HTML formatted text.
– caretPosition (type int): The position of the cursor.
– fullQuote (type boolean): Whether emails created using this template contain the text of the answered or forwarded email as a full quote.
– contentBase64Encoded (type string): The content of the template as a Base64 encoded string.

- **return value:**

– added (type boolean): Indicates whether the template has been added.

A.70   Change a Template

- **path to WSDL file:**
  - [domain] /wsdl/setTemplate.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which a template is to be modified.
  - templateNameToSet (type string): The name of the template to change.
  - isHTML (type boolean): Whether the template is unformatted text or HTML formatted text.
  - caretPosition (type int): The position of the cursor.
  - fullQuote (type boolean): Whether emails created using this template contain the text of the answered or forwarded email as a full quote.
  - contentBase64Encoded (type string): The content of the template as a Base64 encoded string.

- **return value:**
  - set (type boolean): Indicates whether the template was found under the specified name and the changes were saved.

A.71   Rename a template

- **path to WSDL file:**
  - [domain] /wsdl/renameTemplate.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which a template is to be renamed.
  - templateNameToRename (type string): The name of the template to be renamed.
  - newTemplateName (type string): The new name of the template.

- **return value:**
  - renamed (type boolean): Indicates whether the template was found under the specified name and renamed to newTemplateName.

A.72   Delete templates

- **path to WSDL file:**
  - [domain] /wsdl/deleteTemplates.wsdl

- **input parameter:**
  - sessionID (type string): The session ID of the session used to log in with the account for which templates are to be deleted.
  - templateNameToDelete (type string): The name of a template to be deleted. This field must occur at least once and can occur any number of times to specify all templates to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**
  - deleted (type boolean): Specifies that none of the specified templates will exist after the call.

A.73   Retrieve public calendar entries

- **path to WSDL file:**

  - [domain] /wsdl/getPublicCalendarEntries.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account used to retrieve public calendar entries from accounts having the same domain.
    searchString (type string): Optional keywords for a search. If searchString is specified and is not empty, only calendar entries are returned in which all keywords have been found in the summary or location or resources or description field.
    This is especially useful for mobile or low bandwidth clients. entriesFrom (type dateTime): Optional specification of a start time for the beginning of the events denoted by the calendar entries. If this field is specified, only calendar entries whose startTime field is greater than or equal to entriesFrom will be returned.
    This is especially useful for mobile or low bandwidth clients. entriesTo (type dateTime): Optional specification of an end time for the beginning of the events denoted by the calendar entries. If this field is specified, only calendar entries whose startTime field is less than or equal to entriesTo will be returned.
    This is especially useful for mobile or low bandwidth clients.

- **return value:**

  - accountWithPublicCalendarEntries (type accountWithPublicCalendarEntries): An account of the same domain as the querying one, which has at least one public calendar entry. This field can occur any number of times to transfer all accounts that have public calendar entries (equivalent to an array of type accountWithPublicCalendarEntries). The type accountWithPublicCalendarEntries consists of the following fields:

    * accountName (type string): The name of an account of the same domain as the querying one, which has at least one public calendar entry.
    * calendarEntry (type calendarEntry): A public calendar entry of the account named accountName. This field occurs at least once and can otherwise occur as often as required in order to transfer all public calendar entries of this account. The type calendarEntry consists of the following fields:
      · isPublic (type boolean): Indicates whether the entry has been marked as "public".
      · guid (type string): The GUID of the calendar entry.
      · summary (type string): Brief summary of the event denoted by this calendar entry.
      · startTime (type dateTime): The time at which the event denoted by this calendar entry begins.
      · guid (type string): The time at which the event denoted by this calendar entry ends.
      · reminderInMinutesBeforeStart (type int): The number of minutes before startTime to remind of the beginning of the event denoted by this calendar entry. The value of -1 indicates that you do not want to be reminded.
      · reminderRecipient (type string): An e-mail address to which a reminder of this appointment is to be sent at the time specified by reminderInMinutesBeforeStart. This field can occur any number of times to transfer all recipients of the reminder mail (corresponds to an array of type string).
      · location (type string): The location where the event denoted by this calendar entry is to take place.

· `resources` (type `string`): The resources that the event denoted by this calendar entry occupies.
· `description` (type `string`): Detailed description of the event denoted by this calendar entry.
· `rrule` (type `string`): The recurrence rule of the event denoted by this calendar entry in iCalendar format.

A.74   Retrieve a calendar

- **path to WSDL file:**

    - `[domain] /wsdl/getCalendar.wsdl`

- **input parameter:**

    - `sessionID` (type `string`): The session ID of the session used to log in with the account for which calendar entries are to be retrieved.
    - `accountName` (type `string`): The name of an account having the same domain as the querying one whose public calendar entries are to be retrieved.
    Note: If `accountName` is empty, the entire calendar of the querying account will be returned. If `accountName` is not empty, the public calendar entries of the account named `accountName` will be returned, if this account belongs to the same domain as the querying account. `searchString` (type `string`): Optional keywords for a search. If `searchString` is specified and is not empty, only calendar entries are returned in which all keywords have been found in the `summary` or `location` or `resources` or `description` fields.   This is especially useful for mobile or low bandwidth clients. `entriesFrom` (type `dateTime`): Optional specification of a start time for the beginning of the events denoted by the calendar entries. If this field is specified, only calendar entries whose `startTime` field is greater than or equal to `entriesFrom` will be returned. This is especially useful for mobile or low bandwidth clients. `entriesTo` (type `dateTime`): Optional specification of an end time for the beginning of the events denoted by the calendar entries. If this field is specified, only calendar entries whose `startTime` field is less than or equal to `entriesTo` will be returned.
    This is especially useful for mobile or low bandwidth clients.

- **return value:**

    - `calendarEntry` (type `calendarEntry`): An entry in the calendar. This field can occur any number of times to transfer all entries of the calendar (corresponds to an array of type `calendarEntry`). The type `calendarEntry` consists of the following fields:
        * `isPublic` (type `boolean`): Indicates whether the entry has been marked as "public".
        * `guid` (type `string`): The GUID of the calendar entry.
        * `summary` (type `string`): Brief summary of the event denoted by this calendar entry.
        * `startTime` (type `dateTime`): The time at which the event denoted by this calendar entry begins.
        * `endTime` (type `string`): The time at which the event denoted by this calendar entry ends.
        * `reminderInMinutesBeforeStart` (type `int`): The number of minutes before `startTime` to remind of the beginning of the event named this calendar entry. The value of -1 indicates that you do not want to be reminded.
        * `reminderRecipient` (type `string`): An e-mail address to which a reminder of this appointment is to be sent at the time specified by `reminderInMinutesBeforeStart`. This field can occur any number of times to transfer all recipients of the reminder mail (corresponds to an array of type `string`).

* location (type `string`): The location where the event denoted by this calendar entry is to take place.
* resources (type `string`): The resources that the event denoted by this calendar entry occupies.
* description (type `string`): Detailed description of the event denoted by this calendar entry.
* rrule (type `string`): The recurrence rule of the event denoted by this calendar entry in iCalendar format.

A.75   Add a calendar entry

- **path to WSDL file:**
  - [domain] /wsdl/addCalendarEntry.wsdl

- **input parameter:**
  - sessionID (type `string`): The session ID of the session used to log in with the account for which an entry is to be added to the calendar.
  - isPublic (type `boolean`): Whether the calendar entry is public or not.
  - summary (type `string`): Brief summary of the event denoted by this calendar entry.
  - startTime (type `dateTime`): The time at which the event denoted by this calendar entry begins.
  - endTime (type `string`): The time at which the event denoted by this calendar entry ends.
  - reminderInMinutesBeforeStart (type `int`): The number of minutes before `startTime` to remind of the beginning of the event denoted by this calendar entry. The value of -1 indicates that you do not want to be reminded.
  - reminderRecipient (type `string`): An e-mail address to which a reminder of this appointment is to be sent at the time specified by `reminderInMinutesBeforeStart`. This field can occur any number of times to transfer all recipients of the reminder mail (corresponds to an array of type `string`).
  - location (type `string`): The location where the event denoted by this calendar entry is to take place.
  - resources (type `string`): The resources that the event denoted by this calendar entry occupies.
  - description (type `string`): Detailed description of the event denoted by this calendar entry.
  - rrule (type `string`): The recurrence rule of the event denoted by this calendar entry in iCalendar format.

- **return value:**
  - added (type `boolean`): Indicates whether the entry has been added to the calendar.
  - guid (type `string`): The GUID of the added calendar entry.

A.76   Change a calendar entry

- **path to WSDL file:**
  - [domain] /wsdl/setCalendarEntry.wsdl

- **input parameter:**

- sessionID (type string): The session ID of the session used to log in with the account for which a calendar entry is to be modified.
- isPublic (type boolean): Whether the calendar entry is public or not.
- guidToSet (type string): The GUID of the calendar entry.
- summary (type string): Brief summary of the event denoted by this calendar entry.
- startTime (type dateTime): The time at which the event denoted by this calendar entry begins.
- endTime (type string): The time at which the event denoted by this calendar entry ends.
- reminderInMinutesBeforeStart (type int): The number of minutes before startTime to remind of the beginning of the event denoted by this calendar entry. The value of -1 indicates that you do not want to be reminded.
- reminderRecipient (type string): An e-mail address to which a reminder of this appointment is to be sent at the time specified by reminderInMinutesBeforeStart. This field can occur any number of times to transfer all recipients of the reminder mail (corresponds to an array of type string).
- location (type string): The location where the event denoted by this calendar entry is to take place.
- resources (type string): The resources that the event denoted by this calendar entry occupies.
- description (type string): Detailed description of the event denoted by this calendar entry.
- rrule (type string): The recurrence rule of the event denoted by this calendar entry in iCalendar format.

- **return value:**

  - set (type boolean): Indicates whether the entry was found and changed in the calendar.

A.77   Change the "public" property of calendar entries

- **path to WSDL file:**

  - [domain] /wsdl/setPublicStateOfCalendarEntries.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account for whose calendar entries the "public" property is to be changed.
  - isPublic (type boolean): Specifies what the public property should be set to.
  - guidToSet (type string): This GUID specifies a calendar entry whose "public" property is to be set to isPublic. This field must occur at least once and may otherwise occur as often as required in order to transfer all calendar entries whose "public" property is to be set to isPublic (corresponds to an array of type string with at least 1 element).

- **return value:**

  - set (type boolean): Indicates whether the "public" property has been set to isPublic for all specified entries found in the calendar.

A.78   Delete calendar entries

- **path to WSDL file:**

  – [domain] /wsdl/deleteCalendarEntries.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account for which calendar entries are to be deleted.

  – guidToDelete (type string): The GUID of a calendar entry to be deleted. This field must occur at least once and may otherwise occur as often as required to specify all calendar entries to be deleted (corresponds to an array of type string with at least 1 element).

- **return value:**

  – deleted (type boolean): Indicates that none of the specified entries are in the calendar after the call.

A.79   Retrieve public files

- **path to WSDL file:**

  – [domain] /wsdl/getPublicFiles.wsdl

- **input parameter:**

  – sessionID (type string): The session ID of the session used to log in with the account used to retrieve public files from accounts having the same domain. searchString (type string): Optional keywords for a search. If searchString is specified and not empty, only files with all keywords found in the fileName and/or description fields will be returned.
  This is especially useful for mobile or low bandwidth clients.

- **return value:**

  – accountWithPublicFiles (type accountWithPublicFiles): An account of the same domain as the querying one, which has at least one public file. This field can occur any number of times to transfer all accounts that have public files (equivalent to an array of type accountWithPublicFiles).
  The type accountWithPublicFiles consists of the following fields:

    * accountName (type string): The name of an account of the same domain as the querying one, which has at least one public file.
    * file (type file): A public file of the account named accountName. This field occurs at least once and can otherwise occur as often as required in order to transfer all public files of this account. The type file consists of the following fields:
      · isPublic (type boolean): Indicates whether the file has been marked as "public".
      · guid (type string): The GUID of the file.
      · fileName (type string): The name of the file.
      · uploadDateTime (type dateTime): The time when the file was uploaded.
      · fileSizeInByte (type long): The size of the file in bytes.
      · description (type string): Description of the file.

A.80   Retrieve files

- **path to WSDL file:**

  - [domain] /wsdl/getFiles.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account for which files are to be retrieved.

  - accountName (type string): The name of an account having the same domain as the querying one whose public calendar entries are to be retrieved.
    Note: If accountName is empty, all files of the querying account will be returned. If accountName is not empty, the public files of the account named accountName will be returned, if this account belongs to the same domain as the querying account. searchString (type string): Optional keywords for a search. If searchString is specified and not empty, only files with all keywords found in the fileName and/or description fields will be returned.
    This is especially useful for mobile or low bandwidth clients.

- **return value:**

  - file (type file): A file. This field can occur any number of times to transfer all files (corresponds to an array of type file). The type file consists of the following fields:
    * isPublic (type boolean): Indicates whether the file has been marked as "public".
    * guid (type string): The GUID of the file.
    * fileName (type string): The name of the file.
    * uploadDateTime (type dateTime): The time when the file was uploaded.
    * fileSizeInByte (type long): The size of the file in bytes.
    * description (type string): Description of the file.

A.81   Add a file

- **path to WSDL file:**

  - [domain] /wsdl/addFile.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account to which a file is to be uploaded.

  - isPublic (type boolean): Whether the file is public or not.

  - fileName (type string): The name of the file.

  - description (type string): Description of the file.

  - contentBase64Encoded (type string): The contents of the file as a Base64 encoded string.

- **return value:**

  - added (type boolean): Indicates whether the file was uploaded and saved successfully.

  - guid (type string): The GUID of the added file.

  - uploadDateTime (type dateTime): The time that the server saved as the time the file was uploaded.

A.82   Change a file

- **path to WSDL file:**

  - [domain] /wsdl/setFile.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account for which a file is to be modified.
  - isPublic (type boolean): Whether the file is public or not.
  - guidToSet (type string): The GUID of the file.
  - description (type string): Description of the file.

- **return value:**

  - set (type boolean): Indicates whether the file was found and its properties changed.

A.83   Change the "public" property of files

- **path to WSDL file:**

  - [domain] /wsdl/setPublicStateOfFiles.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account for whose files the "public" property is to be changed.
  - isPublic (type boolean): Specifies what the "public" property should be set to.
  - guidToSet (type string): This GUID specifies a file whose "public" property is to be set to isPublic. This field must occur at least once and may otherwise occur as often as required in order to transfer all files whose "public" property is to be set to isPublic (corresponds to an array of type string with at least 1 element).

- **return value:**

  - set (type boolean): Indicates whether the "public" property has been set to isPublic for all specified files whose GUIDs were found.

A.84   Copy files

- **path to WSDL file:**

  - [domain] /wsdl/copyFiles.wsdl

- **input parameter:**

  - sessionID (type string): The session ID of the session used to log in with the account to which the copied files are to be added.
  - accountName (type string): The name of an account having the same domain as the querying one, from which the files are to be copied (this can also be the querying account).
  - guidToCopy (type string): The GUID of a file to be copied. This field must occur at least once and may otherwise occur as often as required to specify all files to be copied (corresponds to an array of type string with at least 1 element).

- **return value:**

– **copied** (type **boolean**): Specifies whether all specified files found and allowed to be copied have been copied.

Note: If the account specified by **accountName** is the querying account, also files that are not public may be copied, otherwise only public files.

A.85   Delete files

- **path to WSDL file:**

  – **[domain] /wsdl/deleteFiles.wsdl**

- **input parameter:**

  – **sessionID** (type **string**): The session ID of the session used to log in with the account for which files are to be deleted.

  – **guidToDelete** (type **string**): The GUID of a file to be deleted. This field must occur at least once and may otherwise occur as often as required to specify all files to be deleted (corresponds to an array of type **string** with at least 1 element).

- **return value:**

  – **deleted** (type **boolean**): Indicates that none of the specified files exist after the call.

A.86   Upload attachments of a mail to file management

- **path to WSDL file:**

  – **[domain]/wsdl/uploadAttachments.wsdl**

- **input parameter:**

  – **sessionID** (type **string**): The session ID of the session used to log in with the account for which attachments of a mail are to be uploaded.

  – **folderID** (type **string**): The ID of the folder of the mail whose attachments are to be uploaded to the file management.

  – **mailID** (type **string**): The ID of the mail whose attachments are to be uploaded to the file management.

  – **attachmentIndex** (type **int**): The index of an attachment to be uploaded to the file management. The indices correspond to the attachments order in the mail source (beginning with 0).
  This field must occur at least once and may otherwise occur as often as required to specify all attachments to be uploaded (corresponds to an array of type **int** with at least 1 element).

- **return value:**

  – **uploaded** (type **boolean**): Indicates that at least one attachment has been uploaded to the file management.

A.87   Add an attachment (in iCalendar format) to the calendar

- **path to WSDL file:**

    – [domain]/wsdl/addAttachmentToCalendar.wsdl

- **input parameter:**

    – `sessionID` (type `string`): The session ID of the session used to log in with the account for which an attachment of a mail is to be added the calendar.
    – `folderID` (type `string`): The ID of the folder of the mail whose attachment is to be added to the calendar.
    – `mailID` (type `string`): The ID of the mail whose attachment is to be added to the calendar.
    – `attachmentIndex` (type `int`): The index of the attachment (in iCalendar format) to be added to the calendar. The index corresponds to the attachments order in the mail source (beginning with 0).

- **return value:**

    – `added` (type `boolean`): Indicates that at least one event has been recognized an added to the calendar.

B   Saving in the "Drafts" folder

To save an email in the `Drafts` folder, the SMTP protocol is used. An email is saved in this folder by sending it to the mail address `save`. Of course, this only works after a previous authentication to the SMTP server.

If an email is sent to the address `save`, it will be saved in the folder `Drafts`. After completing the mail transfer to the server, it replies with `250 [Mail-ID]`, where `[Mail-ID]` is the ID of the stored mail: `250 1471234331-8-1356870558625`

If an existing mail in the `Drafts` folder is to be overwritten, the ID of the mail to be overwritten can be specified as a parameter of the `data` command:

`data 1471234331-8-1356870558625`

If the specified mail is not found, a new file is created. In this case, after completing the mail transfer, the server transmits the ID of the newly created mail file, otherwise the specified one.

C   Composition of mail and file URLs

To transfer the content of mails and files to the WebServiceClient, the HTTP protocol is used.

The URL of an email is composed as follows:

`http(s)://[domain]/mail&sessionID=[session ID]&folderID=[folder ID]&mailID=[mail id]`

However, using this URL, only the readable part of the message body is transmitted as plain text (either the `text/plain` part of the message or, if it does not exist, the text extracted from the `text/html` part).

To request a certain part of a mail, append following to the URL above:

- `&type=plain` To request the raw `text/plain` part of the mail.

- `&type=html` To request the raw `text/html` part of the mail.

- `&type=header` To request the header of the mail.

- `&type=full` To request the complete source text of the mail.

Existing attachments can be requested by specifying its index. To request the first attachment of the mail, append following to the URL above: `&index=0`

Parts of the mail not being treated as attachment, for which a "Content ID" or "Content Location" is specified, can be requested by this "Content ID" or "Content Location":
`&cidl=1314876647-14-1633523650784`

Note: Attachments with a "Content ID" or "Content Location" still can only be requested by specifying the index.

The URL of a file is composed as follows:
`http(s)://[domain]/file&sessionID=[session ID]&accountName=[account name]&guid=[File GUID]`

Note: Both emails and files are usually transmitted in compressed form and must first be decompressed on the client side.

D   Creating a keystore using the Java SDK keytool

If the POP3S, SMTPS, and HTTPS servers are to be offered, in the configuration file the path to a keystore where the certificates for these TLS servers are stored must be specified. In addition, the format and password of the keystore must be specified.

The following describes how to use the Java keytool[6] [7] to generate a keystore.

To do this, in the console you have to change to the directory where the executable keytool file is located. This is usually the "bin" directory of the JDK.

The procedure then is:

```
C:\Program Files\Java\jdk1.7.0\bin> keytool.exe -genkeypair -alias test -validity 36500
-keyalg RSA -keystore Test_Keystore

Enter the keystore password: 123
Enter the password again: 123
What is your first and last name?
[Unknown]: Felix Billau
What is the name of your organizational unit?
[Unknown]: boss
What is the name of your organization?
[Unknown]: FB software
What is the name of your city or town?
[Unknown]: Stuttgart
What is the name of your province or province?
[Unknown]: Baden-Wuerttemberg
What is the country code (two letters) for this unit?
[Unknown]: de
Is CN = Felix Billau, OU = boss, O = FB software, L = Stuttgart, ST = Baden-W? Rtember
g, C = de right?
```

---

[6]http://download.oracle .com/javase/6/docs/technotes/tools/solaris/keytool.html
[7]http://download.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html

```
   [No]: Yes
Enter the password for <test>.
(Enter if password is the same as for keystore):

C:\Program Files\Java\jdk1.7.0\bin>
```

This will give you a keystore file named "Test Keystore", in the format JKS and the password "123".

Note: At the end, the Enter key must be pressed to give the alias "test" the same password as the keystore.

The path to the keystore file can be specified in the configuration file relative to the root directory of the FB-MailService. If the keystore file is in the root directory of the FB-MailService, then the specification of the file name is sufficient here, ie.:

```
Path to keystore file: Test_Keystore
# Either JCEKS, JKS or PKCS12
Keystore format: JKS
Keystore password: 123
```

By specifying such a keystore file, the respective TLS server of the FB-MailService can be offered.