1 Description

2 20 Program Organization Units
   2.1 01 Simple ......................................................... 3
       2.1.1 FbSmtpSendFile (FB) ........................................ 3
       2.1.2 FbSmtpSendMail (FB) ....................................... 6
       2.1.3 FbSmtpSendRam (FB) ....................................... 8
   2.2 02 Advanced ....................................................... 11
       2.2.1 FbSmtpSend (FB) ........................................... 11

3 29 Types
   3.1 eAuthentication (ENUM) .......................................... 15
   3.2 eEncryption (ENUM) ................................................ 15
   3.3 typAttachment (STRUCT) ........................................ 15
   3.4 typMessage (STRUCT) ............................................ 16

4 80 Status
   4.1 Status (GVL) ....................................................... 18
   4.2 eStatusWagoAppMail (ENUM) ................................... 18

5 ParameterList (PARAMS) ............................................ 19

6 VersionHistory (GVL) ................................................ 20

7 Library Reference ..................................................... 21
This document is automatically generated. Because of this, the chapter 30 Visualization is not shown in this document. If you are interested in getting to know more about visualization, we refer to the library manager of e!Cockpit.

Subject to Changes

WAGO Kontakttechnik GmbH Co. KG reserves the right to provide for any alterations or modifications. WAGO Kontakttechnik GmbH Co. KG owns all rights arising from the granting of patents or from the legal protection of utility patents. Third-party products are always mentioned without any reference to patent rights. Thus, the existence of such rights cannot be excluded.

Personnel Qualification

All tasks that are carried out with libraries made for the e!COCKPIT software must only be performed by qualified electrical specialists instructed in PLC programming according to IEC 61131-3.

All tasks that have an effect on the properties or the behavior of automation hardware or software products must only be performed by qualified employees with a thorough knowledge of handling the products concerned.

Intended Use of e!COCKPIT Libraries

Libraries created for the e!COCKPIT software are used to simplify the development of application projects in the IEC 61131-3 programming languages.

For automation tasks, WAGO offers programmable logic controllers in a wide variety of performance classes. In combination with a wide range of I/O modules, the controllers can process standard types of field signals. Controllers can be implemented centrally or in decentralized configurations. The controllers offer interfaces for the most commonly used fieldbuses for use in decentralized configurations. Fieldbus independent I/O modules are then linked via fieldbus couplers. WAGO controllers offer a runtime environment for user programs called e!RUNTIME. Software projects for implementation in e!RUNTIME environments can be created in e!COCKPIT. The programming environment in e!COCKPIT is based on the established CODESYS 3 industrial standard. Users with a previous knowledge of CODESYS 3 will thus find this environment largely familiar. The following programming languages of the IEC 61131-3 standard are available:

- Structured Text (ST)
- Ladder Diagram (LD)
- Function Block Diagram (FBD)
- Instruction List (IL)
- Sequential Function Chart (SFC)
- Continuous Function Chart (CFC)

The individual programming languages can also be combined as required during the development of the software. A portfolio of prepared libraries can be accessed for many frequently used functions in order to make software development more efficient. This document provides an overview of the WagoAppMail that WAGO offers for e!COCKPIT.
This library implements SMTP client functionalities.

Further library information are summarized here:

- **Company**: WAGO
- **Title**: WagoAppMail
- **Version**: 1.1.1.1
- **Categories**: Application; WAGO LayerView/App; WAGO FunctionalView/Connectivity
- **Namespace**: WagoAppMail
- **Author**: WAGO / u13773
- **Placeholder**: WagoAppMail
2.1 01 Simple

Straightforward functions blocks to send plain text email messages. Optional support for attachments from filesystem or from data array.

2.1.1 FbSmtpSendFile (FB)

Interface variables
<table>
<thead>
<tr>
<th>Scope</th>
<th>Name</th>
<th>Type</th>
<th>Comment</th>
<th>Inherited from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inout</td>
<td>xTrigger</td>
<td>BOOL</td>
<td>Triggers the execution of the action.</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td>Output</td>
<td>xBusy</td>
<td>BOOL</td>
<td>Action is still in progress.</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>xError</td>
<td>BOOL</td>
<td>Indicates that an error has occurred.</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>oStatus</td>
<td>WagoSysError-Base.FbResult</td>
<td>Status object (see WagoSysErrorBase).</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td>Input</td>
<td>sServer</td>
<td>STRING</td>
<td>Servername or IP address of SMTP server: e.g. 192.168.1.25 or smtp.gmail.com</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>wPort</td>
<td>WORD</td>
<td>Port number of SMTP server</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sUser</td>
<td>WagoApp-String.MaxString</td>
<td>Username of SMTP account</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sPassword</td>
<td>WagoApp-String.MaxString</td>
<td>Password of SMTP account</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>eAuthentication</td>
<td>eAuthentication</td>
<td>Authentication method</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>eEncryption</td>
<td>eEncryption</td>
<td>Encryption type</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>typSSL_Options</td>
<td>WagoTypesCurl.typSSL_Options</td>
<td>Options for SSL connection (see WagoTypesCurl)</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>tTimeout</td>
<td>TIME</td>
<td>If tTimeout = 0, the timeout is deactivated</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sFrom</td>
<td>WagoApp-String.MaxString</td>
<td>FROM address</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sTo</td>
<td>WagoApp-String.MaxString</td>
<td>TO address(es): Multiple addresses must be separated with semicolon: <a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sCc</td>
<td>WagoApp-String.MaxString</td>
<td>CC address(es): Multiple addresses must be separated with semicolon: <a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sBcc</td>
<td>WagoApp-String.MaxString</td>
<td>BCC address(es): Multiple addresses must be separated with semicolon: <a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sSubject</td>
<td>WagoApp-String.MaxString</td>
<td>Subject text</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sMessage</td>
<td>WagoApp-String.MaxString</td>
<td>Message text</td>
<td>Fb-SmtpSendMail</td>
</tr>
<tr>
<td></td>
<td>sAttachment</td>
<td>WagoApp-String.MaxString</td>
<td>Path of file that should be attached: e.g. /var/log/messages</td>
<td>Fb-SmtpSendMail</td>
</tr>
</tbody>
</table>
Function

Send plain text email messages with attachment from filesystem via encrypted or unencrypted SMTP.

Graphical Illustration

<table>
<thead>
<tr>
<th>FunctionBlock</th>
</tr>
</thead>
<tbody>
<tr>
<td>FbSmtpSendFile</td>
</tr>
</tbody>
</table>

Function description

This function block allows to send email messages with attachment from filesystem via encrypted or unencrypted SMTP. The message body of the email is configurable through the input variable `sMessage`. The input variable `sMessage` contains the path to the file that should be attached to the email message.

Transition to TRUE on `xTrigger` triggers the process to create and send an email message. The function block resets `xTrigger` to FALSE again after it has finished the process. If something went wrong during the process `xTrigger` resets to FALSE and `xError` is set to TRUE. The output `xBusy` indicates that the function block is still processing the email message.

Example  Send a simple plain text email message with attachment from filesystem:

```plaintext
VAR
  oSmtpSend : WagoAppMail.FbSmtpSendFile;
  xTrigger : BOOL;
  oStatus : WagoAppMail.WagoSysErrorBase.FbResult;
  xError : BOOL;
  xBusy : BOOL;
  sServer : STRING := '192.168.1.57';
  wPort : WORD := 25;
  sUser : STRING := 'test1@application.wago';
  sPwd : STRING := 'wago';
  eAuthentication := WagoAppMail.eAuthentication.eLOGIN;
  eEncryption := WagoAppMail.eEncryption.eUNENCRYPTED;
  typSSL_Options : WagoAppMail.WagoTypesCurl.typSSL_Options;
  tTimeout : TIME;
  sFrom : STRING := 'test1@application.wago';
  sTo : STRING := 'test2@application.wago;test3@application.wago';
  sCc : STRING := 'test1@application.wago';
  sBcc : STRING := '';
  sSubject : STRING := 'This message is send via WagoAppMail';
```
WagoAppMail, Release 1.1.1.1

2.1.2 FbSmtpSendMail (FB)

Interface variables
### Function

Send plain text email messages via encrypted or unencrypted SMTP.

#### Graphical Illustration

![FunctionBlock](image)

#### Function description

2.1. 01 Simple
This function block allows to send email messages via encrypted or unencrypted SMTP. The message body of the email is configurable through the input variable \texttt{sMessage}.

Transition to TRUE on \texttt{xTrigger} triggers the process to create and send an email message. The function block resets \texttt{xTrigger} to FALSE again after it has finished the process. If something went wrong during the process \texttt{xTrigger} resets to FALSE and \texttt{xError} is set to TRUE. The output \texttt{xBusy} indicates that the function block is still processing the email message.

**Example** Send a simple plain text email message:

```plaintext
VAR
  oSmtpSend : WagoAppMail.FbSmtpSendFile;
  xTrigger : BOOL;
  oStatus : WagoAppMail.WagoSysErrorBase.FbResult;
  xError : BOOL;
  xBusy : BOOL;
  sServer : STRING := '192.168.1.57';
  wPort : WORD := 25;
  sUser : STRING := 'test1@application.wago';
  sPwd : STRING := 'wago';
    \(---> eAuthentication.eLOGIN;
    \(---> eUNENCRYPTED;
  typSSL_Options : WagoAppMail.WagoTypesCurl.typSSL_Options;
  tTimeout : TIME;
  sFrom : STRING := 'test1@application.wago';
  sTo : STRING := 'test2@application.wago;test3@application.wago';
  sCc : STRING := 'test1@application.wago';
  sBcc : STRING := '';
  sSubject : STRING := 'This message is send via WagoAppMail';
  sMessage : STRING := 'This is the message text send via WagoAppMail';
END_VAR

oSmtpSend(
  xTrigger:= xTrigger,
  oStatus=> oStatus,
  xError=> xError,
  xBusy=> xBusy,
  sServer:= sServer,
  wPort:= wPort,
  sUser: sUser,
  sPassword: sPwd,
  eAuthentication:= eAuthentication,
  eEncryption:= eEncryption,
  typSSL_Options:= typSSL_Options,
  tTimeout:= tTimeout,
  sFrom:= sFrom,
  sTo:= sTo,
  sCc:= sCc,
  sBcc:= sBcc,
  sSubject:= sSubject,
  sMessage:= sMessage,
  sAttachment:= sAttachment);
```

**2.1.3 FbSmtpSendRam (FB)**

**Interface variables**
<table>
<thead>
<tr>
<th>Scope</th>
<th>Name</th>
<th>Type</th>
<th>Comment</th>
<th>Inherited from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inout</td>
<td>xTrigger</td>
<td>BOOL</td>
<td>Triggers the execution of the action.</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>xBusy</td>
<td>BOOL</td>
<td>Action is still in progress.</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>xError</td>
<td>BOOL</td>
<td>Indicates that an error has occurred.</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>oStatus</td>
<td>WagoSysEr-</td>
<td>Status object (see WagoSysErrorBase).</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ror-Base.FbResult</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>sServer</td>
<td>STRING</td>
<td>Servername or IP address of SMTP server: e.g. 192.168.1.25 or smtp.gmail.com</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>wPort</td>
<td>WORD</td>
<td>Port number of SMTP server</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>sUser</td>
<td>WagoApp-</td>
<td>Username of SMTP account</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sPassword</td>
<td>WagoApp-</td>
<td>Password of SMTP account</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>eAuthenti-</td>
<td>cation</td>
<td>Authentication method</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>eEncryption</td>
<td></td>
<td>Encryption type</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>typSSL_Option</td>
<td>WagoTypes-</td>
<td>Options for SSL connection (see WagoTypesCurl)</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Curl.typSSL_Options</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>tTimeout</td>
<td>TIME</td>
<td>If tTimeout = 0, the timeout is deactivated</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td>sFrom</td>
<td>WagoApp-</td>
<td>FROM address</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sTo</td>
<td>WagoApp-</td>
<td>TO address(es): Multiple addresses must be separated with semicolon:</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td><a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sCc</td>
<td>WagoApp-</td>
<td>CC address(es): Multiple addresses must be separated with semicolon:</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td><a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sBcc</td>
<td>WagoApp-</td>
<td>BCC address(es): Multiple addresses must be separated with semicolon:</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td><a href="mailto:address1@test.org">address1@test.org</a>;<a href="mailto:address2@test.org">address2@test.org</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sSubject</td>
<td>WagoApp-</td>
<td>Subject text</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sMessage</td>
<td>WagoApp-</td>
<td>Message text</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sAttachmentName</td>
<td></td>
<td>Name of attachment</td>
<td>Fb-SmtpSend-Mail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WagoApp-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>String.MaxString</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pData</td>
<td>POINTER TO</td>
<td>Address of the data that should be attached to the email</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYTE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>uSize</td>
<td>UDINT</td>
<td>Size of the data that should be attached</td>
<td></td>
</tr>
</tbody>
</table>
Function
Send plain text email messages with attachment data taken directly from the e!COCKPIT application via encrypted or unencrypted SMTP.

Graphical Illustration

Function description
This function block allows to send plain text email messages with attachment data taken directly from the e!COCKPIT application via encrypted or unencrypted SMTP. The message body of the email is configurable through the input variable `sMessage`. `sAttachmentName` determines the name of the attachment in the email message. The inputs `pData` and `udiSize` are determining the data that is attached to the email message.

Transition to TRUE on `xTrigger` triggers the process to create and send an email message. The function block resets `xTrigger` to FALSE again after it has finished the process. If something went wrong during the process `xTrigger` resets to FALSE and `xError` is set to TRUE. The output `xBusy` indicates that the function block is still processing the email message.

Example Send a simple plain text email message with attachment data taken from data array:

```c
VAR
  oSmtpSend : WagoAppMail.FbSmtpSendRam;
  xTrigger : BOOL;
  oStatus : WagoAppMail.WagoSysErrorBase.FbResult;
  xError : BOOL;
  xBusy : BOOL;
  sServer : STRING := '192.168.1.57';
  wPort : WORD := 25;
  sUser : STRING := 'test1@application.wago';
  sPwd : STRING := 'wago';
  eAuthentication : WagoAppMail.eAuthentication := WagoAppMail.eAuthentication.eLOGIN;
  eEncryption : WagoAppMail.eEncryption := WagoAppMail.eEncryption.eUNENCRYPTED;
  typSSL_Options : WagoAppMail.WagoTypesCurl.typSSL_Options;
  tTimeout : TIME;
  sFrom : STRING := 'test1@application.wago';
  sTo : STRING := 'test2@application.wago;test3@application.wago';
  sSubject : STRING := 'Simple Message';
  sMessage : MaxString := 'This is a simple message.';
  sAttachmentName : MaxString := 'attachment';
  pData : Pointer;
  udiSize : UDINT;
```
2.2 02 Advanced

Advanced function block to send email messages with configurable message body and attachments.

2.2.1 FbSmtpSend (FB)

Interface variables
### Function

Send email messages via encrypted or unencrypted SMTP.

### Graphical Illustration

![Function Block Diagram]

```plaintext
FunctionBlock FbsMtpSend

xTrigger BOOL xBusy
sServer STRING wPort
sUser WagoAppString MaxString FbResult oStatus
sPassword WagoAppString MaxString
eAuthentication eEncryption
typSSL_Options WagoTypesCurl.typSSL_Options
fTimeout TIME
sFrom WagoAppString MaxString
sTo WagoAppString MaxString
sCc WagoAppString MaxString
sBcc WagoAppString MaxString
sSubject WagoAppString MaxString
typMessage typMessage
aAttachment POINTER TO typAttachment
```
Function description

This function block allows to send email messages via encrypted or unencrypted SMTP. The message body of the email is configurable through the input variable `typMessage (STRUCT)`. The structure `typMessage (STRUCT)` allows to configure the data area, size, content-type, character set and encoding of the message body. This may be helpful if it is required to send HTML content in the message body (text/html).

The number of attachments is not limited by the function block. Limiting factors are the available space in the tmp-filesystem of the controller and restrictions of the email provider.

Attachments are configurable through the input array `typAttachment (STRUCT)`. Each element of the array holds an description of an attachment. The attachment may be a file located in the filesystem of the controller or a data array declared inside the e!COCKPIT application.

Transition to TRUE on `xTrigger` triggers the process to create and send an email message. The function block resets `xTrigger` to FALSE again after it has finished the process. If something went wrong during the process `xTrigger` resets to FALSE and `xError` is set to TRUE. The output `xBusy` indicates that the function block is still processing the email message.

Example  Send an email message with attachments taken from filesystem and from e!COCKPIT data array:

```plaintext
VAR
  oSmtpSend : WagoAppMail.FbSmtpSend;
  xTrigger : BOOL;
  oStatus : WagoAppMail.WagoSysErrorBase.FbResult;
  xError : BOOL;
  xBusy : BOOL;
  sServer : STRING := '192.168.1.57';
  wPort : WORD := 25;
  sUser : STRING := 'test1@application.wago';
  sPwd : STRING := 'wago';
  eAuthentication : WagoAppMail.eAuthentication := WagoAppMail.eAuthentication.eLOGIN;
  eEncryption : WagoAppMail.eEncryption := WagoAppMail.eEncryption.eUNENCRYPTED;
  typSSL_Options : WagoAppMail.WagoTypesCurl.typSSL_Options;
  tTimeout :
  sFrom : STRING := 'test1@application.wago';
  sTo :
    sCc :
    sBcc :
  sSubject :
  typMessage : WagoAppMail.typMessage;
  aAttachment :
    aData :
      sMyMessage :
        typMessage.pData := ADR(sMyMessage);
        typMessage.udiSize := TO_UDINT(LEN(sMyMessage));
        typMessage.sContentType := 'text/plain';
        typMessage.sCharset := 'windows-1252';
END_VAR
```
typMessage.sEncoding := '8bit';
aAttachment[1].sFilename := 'LogMessages.txt';
aAttachment[1].sPath := '/var/log/messages';
aAttachment[1].pData := 0;
aAttachment[1].udiSize := 0;
aAttachment[2].sFilename := 'MyPlcData.txt';
aAttachment[2].sPath := '';
aAttachment[2].pData := ADR(aData);
aAttachment[2].udiSize := SIZEOF(aData);

oSsmtpSend(
    xTrigger:= xTrigger,
    oStatus=> oStatus,
    xError=> xError,
    xBusy=> xBusy,
    sServer:= sServer,
    wPort:= wPort,
    sUser:= sUser,
    sPassword:= sPwd,
    eAuthentication:= eAuthentication,
    eEncryption:= eEncryption,
    typSSL_Options:= typSSL_Options,
    tTimeout:= tTimeout,
    sFrom:= sFrom,
    sTo:= sTo,
    sCc:= sCc,
    sBcc:= sBcc,
    sSubject:= sSubject,
    typMessage:= typMessage,
    aAttachment:= aAttachment);
### 3.1 eAuthentication (ENUM)

<table>
<thead>
<tr>
<th>Name</th>
<th>Initial</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>eAUTOMATIC</td>
<td>0</td>
<td>Select authentication method automatically. The function block may select any mechanism supported by the server. This option should be treated with care as it might cause a fall back to a weak security mechanism.</td>
</tr>
<tr>
<td>eNONE</td>
<td>1</td>
<td>No authentication</td>
</tr>
<tr>
<td>ePLAIN</td>
<td>2</td>
<td>Plain</td>
</tr>
<tr>
<td>eLOGIN</td>
<td>3</td>
<td>Login</td>
</tr>
<tr>
<td>eCGRAM_MD5</td>
<td>4</td>
<td>Cram-MD5</td>
</tr>
<tr>
<td>eDIGEST_MD5</td>
<td>6</td>
<td>Digest-MD5</td>
</tr>
<tr>
<td>eNTLM</td>
<td>7</td>
<td>NTLM</td>
</tr>
</tbody>
</table>

### 3.2 eEncryption (ENUM)

<table>
<thead>
<tr>
<th>Name</th>
<th>Initial</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>eSTARTTLS</td>
<td>0</td>
<td>Open an unencrypted TCP connection to the SMTP server and switch to TLS afterwards</td>
</tr>
<tr>
<td>eSMTPS</td>
<td>1</td>
<td>Open an encrypted connection directly</td>
</tr>
<tr>
<td>eUNENCRYPTED</td>
<td>2</td>
<td>No encryption</td>
</tr>
</tbody>
</table>

### 3.3 typAttachment (STRUCT)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>sFilename</td>
<td>Wago-Types.FileName</td>
<td>This name is used for the attachment in the email message</td>
</tr>
<tr>
<td>sPath</td>
<td>Wago-Types.FileName</td>
<td>Path of the file that should be attached to the email message: e.g. /var/log/messages</td>
</tr>
<tr>
<td>pData</td>
<td>POINTER TO BYTE</td>
<td>Pointer to the data that should be attached to the email message. This parameter is only considered if sPath is not set</td>
</tr>
<tr>
<td>udiSize</td>
<td>UDINT</td>
<td>Size of the data area that is referenced by pData</td>
</tr>
</tbody>
</table>
This structure describes a single attachment element. Each attachment may be a file located in the file system or a data area declared in the e!COCKPIT application.

```plaintext
VAR
  aAttachment : ARRAY [1..5] OF typAttachment :=
    [
      //This element attaches the file /home/codesys/data.txt
      (sFilename := '',
       sPath := '/home/codesys/data.txt',
       pData := 0,
       udiSize := 0),
      //This element attaches the file /home/codesys/data.txt
      and renames it to MyDataFile.dat
      (sFilename := 'MyDataFile.dat',
       sPath := '/home/codesys/data.txt',
       pData := 0,
       udiSize := 0),
      //This element attaches the data referenced by aData and
      names the file MyDataFile.dat
      (sFilename := 'MyDataFile.dat',
       sPath := '',
       pData := ADR(aData),
       udiSize := SIZEOF(aData)),
      //This element attaches the file /home/codesys/data.txt
      without renaming it, pData and udiSize are ignored
      (sFilename := '',
       sPath := '/home/codesys/data.txt',
       pData := ADR(aData),
       udiSize := SIZEOF(aData)),
      //This element causes an error, because sFilename must be
      set for data attachments
      (sFilename := '',
       sPath := '',
       pData := ADR(aData),
       udiSize := SIZEOF(aData))
    ]
END_VAR
```

3.4 typMessage (STRUCT)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Initial</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>pData</td>
<td>POINTER TO BYTE</td>
<td>Pointer to the data area that holds the message</td>
<td></td>
</tr>
<tr>
<td>udiSize</td>
<td>UDINT</td>
<td>Size of the data area that is referenced by pData</td>
<td></td>
</tr>
<tr>
<td>sContentType</td>
<td>STRING</td>
<td>‘text/plain’</td>
<td>Content-Type of the message</td>
</tr>
<tr>
<td>sCharset</td>
<td>STRING</td>
<td>‘windows-1252’</td>
<td>Charset of the message</td>
</tr>
<tr>
<td>sEncoding</td>
<td>STRING</td>
<td>‘8bit’</td>
<td>Content-Transfer-Encoding of the message</td>
</tr>
</tbody>
</table>

This structure describes the message body. It allows to specify the data area and the size of the body data. In
addition it is possible to set the header field content-type, character set and encoding.
The default header of the message body includes the following parameter:

Content-Type: text/plain; charset="windows-1252"
Content-Transfer-Encoding: 8bit

<Data referenced by pData is inserted here>

If the header fields are modified (e.g. text/html instead of text/plain) the message data (pData) has to be provided in the correct form, as the function block simply copies the data to the message without interpreting or encoding.
4.1 Status (GVL)

<table>
<thead>
<tr>
<th>Scope</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>gc_StatusWagoAppMail</td>
<td>ARRAY [0..3] OF WagoSysErrorBase.WagoTypesErrorBase.typResultItem</td>
</tr>
</tbody>
</table>

**Description:** Status Information

<table>
<thead>
<tr>
<th>Value</th>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>WagoSysErrorBase.WagoTypes.eSeverity.none</td>
<td>‘OK’</td>
</tr>
<tr>
<td>1</td>
<td>WagoSysErrorBase.WagoTypes.eSeverity.error</td>
<td>‘No valid message interface available’</td>
</tr>
<tr>
<td>2</td>
<td>WagoSysErrorBase.WagoTypes.eSeverity.error</td>
<td>‘Attachment is not available’</td>
</tr>
<tr>
<td>3</td>
<td>WagoSysErrorBase.WagoTypes.eSeverity.error</td>
<td>‘Read attachment data has failed’</td>
</tr>
</tbody>
</table>

4.2 eStatusWagoAppMail (ENUM)

<table>
<thead>
<tr>
<th>Name</th>
<th>Initial</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ok</td>
<td>0</td>
<td>(none) Ok</td>
</tr>
<tr>
<td>UndefinedInterface</td>
<td>1</td>
<td>Interfaces are not defined</td>
</tr>
<tr>
<td>AttachmentNotAvailable</td>
<td>2</td>
<td>Attachment is not available</td>
</tr>
<tr>
<td>ReadAttachmentFailed</td>
<td>3</td>
<td>Read attachment data failed</td>
</tr>
</tbody>
</table>
## ParameterList (PARAMS)

<table>
<thead>
<tr>
<th>Scope</th>
<th>Name</th>
<th>Type</th>
<th>Initial</th>
<th>Comment</th>
</tr>
</thead>
</table>
| Constant  | DELETE_TEMPFILES | BOOL  | TRUE    | Configure handling of temporary files generated by FbMessageHandler (Files are located in /tmp) 
TRUE : Delete temporary files after email is sent 
FALSE : Do not delete temporary files (Maybe useful for debugging) |
| InOut:    | CURL_LOGGING   | BOOL  | FALSE   | Configure verbose mode and logging of FbCurl 
TRUE : Enable verbose mode and logging to /var/log/WagoAppMail_Instance_x (Files are not rotated automatically!)
FALSE : Disable logging |
## Version History (GVL)

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Author</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.07.2019</td>
<td>1.1.1.1</td>
<td>WAGO / u013972</td>
<td>Changes for new codesys version</td>
</tr>
<tr>
<td>08.01.2019</td>
<td>1.1.1.0</td>
<td>u015842</td>
<td>Properties: free placeholder added</td>
</tr>
<tr>
<td>15.05.2018</td>
<td>1.1.0.6</td>
<td>WAGO / u013773</td>
<td>Add parameter to configure logging</td>
</tr>
<tr>
<td>27.04.2018</td>
<td>1.1.0.5</td>
<td>WAGO / u010545</td>
<td>Update documentation</td>
</tr>
<tr>
<td>16.10.2017</td>
<td>1.1.0.4</td>
<td>WAGO / u013773</td>
<td>Placeholder for WagoAppFileDir modified</td>
</tr>
<tr>
<td>05.09.2017</td>
<td>1.1.0.3</td>
<td>WAGO / u013773</td>
<td>Use CURL_NULL instead of empty string if user/password is not set</td>
</tr>
<tr>
<td>07.04.2017</td>
<td>1.1.0.2</td>
<td>WAGO / u013773</td>
<td>Use local time instead of UTC for time header</td>
</tr>
<tr>
<td>07.04.2017</td>
<td>1.1.0.1</td>
<td>WAGO / u013773</td>
<td>Newline indicator within attachment data block corrected</td>
</tr>
<tr>
<td>15.12.2016</td>
<td>1.1.0.0</td>
<td>WAGO / u013773</td>
<td>Variable names harmonized</td>
</tr>
<tr>
<td>16.11.2016</td>
<td>1.0.0.0</td>
<td>WAGO / u013773</td>
<td>First release</td>
</tr>
</tbody>
</table>

**Release Notes:** Firmware version >= 02.06.xx(09) for PFC100/PFC200 required

**Known Bugs:**
This is a dictionary of all referenced libraries and their name spaces.

**Util**

*Library Identification:*
- Placeholder: Util
- Default Resolution: Util, * (System)
- Namespace: Util

*Library Properties:*
- LinkAllContent: False
- Optional: False
- HideWhenReferencedAsDependency: True
- QualifiedOnly: True
- SystemLibrary: False

*Library Parameter:*
- Parameter: IBLOCKSIZE = 22800

**WagoAppFileDir**

*Library Identification:*
- Placeholder: WagoAppFileDir
- Default Resolution: WagoAppFileDir, * (WAGO)
- Namespace: WagoAppFileDir

*Library Properties:*
- LinkAllContent: False
- QualifiedOnly: False
- SystemLibrary: False
- Optional: False

*Library Parameter:*
- Parameter: STANDARDSCHEDULINGMODE = eSchedulingMode.Background
- Parameter: STANDARDTIMEOUT = TIME#10s0ms
WagoAppMail, Release 1.1.1.1

### WagoAppString
*Library Identification:*
Placeholder: WagoAppString
Default Resolution: WagoAppString, * (WAGO)
Namespace: WagoAppString

*Library Properties:*
- LinkAllContent: False
- QualifiedOnly: False
- SystemLibrary: False
- Optional: False

### WagoAppTime
*Library Identification:*
Placeholder: WagoAppTime
Default Resolution: WagoAppTime, * (WAGO)
Namespace: WagoAppTime

*Library Properties:*
- LinkAllContent: False
- QualifiedOnly: False
- SystemLibrary: False
- Optional: False

### WagoSysCurl
*Library Identification:*
Placeholder: WagoSysCurl
Default Resolution: WagoSysCurl, * (WAGO)
Namespace: WagoSysCurl

*Library Properties:*
- LinkAllContent: False
- QualifiedOnly: False
- SystemLibrary: False
- Optional: False

### WagoSysErrorBase
*Library Identification:*
Placeholder: WagoSysErrorBase
Default Resolution: WagoSysErrorBase, * (WAGO)
Namespace: WagoSysErrorBase

*Library Properties:*
- LinkAllContent: False
- QualifiedOnly: False
- SystemLibrary: False
- Optional: False
• LinkAllContent: False
• QualifiedOnly: True
• SystemLibrary: False
• Optional: False

Library Parameter:
Parameter: RES_LOG_MAX_FILESIZE = 2000
Parameter: RES_LOG_MAX_FILES = 1
Parameter: RES_LOG_MAX_ENTRIES = 200
Parameter: RES_LOG_NAME = ‘WagoAppResultLogger’

WagoSysPlainMem
Library Identification:
Placeholder: WagoSysPlainMem
Default Resolution: WagoSysPlainMem, * (WAGO)
Namespace: WagoSysPlainMem

Library Properties:

• LinkAllContent: False
• QualifiedOnly: False
• SystemLibrary: False
• Optional: False

WagoSysVersion
Library Identification:
Name: WagoSysVersion
Version: 1.0.0.0
Company: WAGO
Namespace: WagoSysVersion

Library Properties:

• LinkAllContent: False
• QualifiedOnly: True
• SystemLibrary: False
• Optional: False

WagoTypesCommon
Library Identification:
Placeholder: WagoTypesCommon
Default Resolution: WagoTypesCommon, * (WAGO)
Namespace: WagoTypes

Library Properties:

• LinkAllContent: False
WagoAppMail, Release 1.1.1.1

- Optional: False
- QualifiedOnly: False
- SystemLibrary: False
- PublishSymbolsInContainer: True

WagoTypesCurl

Library Identification:
Placeholder: WagoTypesCurl
Default Resolution: WagoTypesCurl, * (WAGO)
Namespace: WagoTypesCurl

Library Properties:

- LinkAllContent: False
- Optional: False
- QualifiedOnly: False
- SystemLibrary: False
- PublishSymbolsInContainer: True

© WAGO Kontakttechnik GmbH & Co. KG, Germany 2018 – All rights reserved. For the avoidance of doubt, this copyright notice does not only apply to the information above but also and primarily to the described library itself. Please note that third-party products are always mentioned without reference to intellectual property rights, including patents, utility models, designs and trademarks, accordingly the existence of such rights cannot be excluded. WAGO is a registered trademark of WAGO Verwaltungsgesellschaft mbH.