



Mailen mit Konzept 16

Mit Konzept 16 ist es besonders einfach, ereignisgesteuerte E-Mails zu versenden. Dazu muß nur mit drei Funktionen gearbeitet werden.

von Florian Lapp

Concept 16 stellt für den E-Mail-Versand das Mail-Objekt zur Verfügung. Es übernimmt die Kommunikation mit dem Mailserver. Über das SMTP-Protokoll, über das der Versand läuft, braucht man sich nicht zu kümmern, für diese Ausgabe ist das Mail-Objekt zuständig. Die E-Mail wird mit Hilfe des Mail-Objekts erzeugt, zusammengesetzt und versendet. Dafür stehen die drei Funktionen *MailOpen*, *MailData* und *MailClose* zur Verfügung. Das E-Mail-Objekt wird eingangs mit einem Aufruf von *MailOpen()* angelegt:

```
tMail # MailOpen(_MailSMTP,           // E-Mail-Typ
          'smtp.mymailserver.com', // Server-Name
          25,                          // Server-Port
          0,
          'myusername',                 // Benutzername
          'mypassword');               // Kennwort
```

Anschließend wird mit Aufrufen von *MailData()* der Nachrichtenheader definiert:

```
tMail->MailData(_SMTPFrom, 'f.lapp@vectorsoft.de');
tMail->MailData(_SMTPTo, 'f.lapp@vectorsoft.de');

// Absender definieren
tMail->MailData(_SMTPFrom, 'tom@work.org', 'Müller, Tom');
// Empfänger definieren
tMail->MailData(_SMTPTo, 'tina@home.net', 'Schmidt, Tina');
tMail->MailData(_SMTPBcc, 'emil@school.com', 'Meier, Emil');
tMail->MailData(_SMTPBcc, 'lea@studio.biz', 'Schneider,
               Lea');
// Betreff definieren
tMail->MailData(_SMTPSubject, 'Warnung: Verfügbare ' +
               'Festplattenkapazität unter 10 GByte');
```

Der eigentliche Inhalt der E-Mail, der E-Mail-Text, kann entweder als reiner Text definiert oder als HTML-Mail formatiert werden. Der einfachste Typ einer E-Mail ist die Text-Mail. Sie besteht nur aus ASCII-Zeichen und enthält keinerlei Formatierungen. Sie wird auch von jedem Zielbrowser verarbeitet:

```
tMail->MailData(_MailLine,
               'Die verfügbare Festplattenkapazität ' +
               'beträgt weniger als 10 GByte. ');
tMail->MailData(_MailLine, '');
tMail->MailData(_MailLine, 'Festplattenkapazität');
tMail->MailData(_MailLine, 'Gesamt: 200 GByte');
```

MailOpen

Die Funktion *MailOpen* erzeugt ein Mail-Objekt:

```
MailOpen(
  MailType      : int;    // E-Mail-Typ
  Address       : alpha;  // Server-Name
  opt Port      : int;    // Server-Port
  opt Timeout   : int;    // Timeout (Sekunden)
  opt User      : alpha;  // Benutzername
  opt Password  : alpha;  // Kennwort
) : handle          // Objekt/Fehler
```

Als E-Mail-Typ muß der Entwickler *_MailSMTP* angeben. Dabei ist der Server-Port zu übergeben, falls er vom Standardport 25 abweicht. Zusätzlich kann ein Timeout in Sekunden für die Kommunikation definiert werden, der Standardwert für das Timeout ist 60 Sekunden. Zur Authentifizierung beim Mailserver ist es zudem möglich, Benutzername und Kennwort zu übergeben.

```
tMail->MailData(_MailLine, 'Belegt: 191,26 GByte');
tMail->MailData(_MailLine, 'Verfügbar: 8,74 GByte');
```

Neben solchen einfachen Text-Mails besteht die Möglichkeit, den Text mit HTML zu formatieren. Der HTML-Text kann auch zusätzlich zum unformatierten Text definiert werden. Der Mail-Client entscheidet dann, welche der beiden Varianten er darstellt:

```
tMail->MailData(_MailLine | _MimeTextHTML, '<html>');
tMail->MailData(_MailLine | _MimeTextHTML, '<body>');
tMail->MailData(_MailLine | _MimeTextHTML,
               'Die verfügbare Festplattenkapazität ' +
               'beträgt weniger als 10 GByte.<br> ');
tMail->MailData(_MailLine | _MimeTextHTML, '</br>');
tMail->MailData(_MailLine | _MimeTextHTML,
               '<table border="1" style="font-weight:bold"> ');
tMail->MailData(_MailLine | _MimeTextHTML,
               '<tr><td colspan="2">Festplattenkapazität</td></tr> ');
tMail->MailData(_MailLine | _MimeTextHTML,
               '<tr><td>Gesamt</td><td>200GB</td></tr> ');
tMail->MailData(_MailLine | _MimeTextHTML,
               '<tr><td>Belegt</td><td>191,26GB</td></tr> ');
tMail->MailData(_MailLine | _MimeTextHTML,
               '<tr><td>Verfügbar</td> ' +
               '<td style="color:red">8,74GB</td></tr>');

tMail->MailData(_MailLine | _MimeTextHTML, '</table>');
tMail->MailData(_MailLine | _MimeTextHTML, '</body>');
```

MailData

MailData definiert die Daten der E-Mail:

```
MailData(
    MailHdl      : handle; // Mail-Objekt
    Type         : int;    // Datentyp
    Value1       : alpha;  // Wert 1
    opt Value2   : alpha;  // Wert 2
) : int          // Ergebnis der Funktion
```

Die folgenden Datentypen stehen für den E-Mail-Kopf zur Verfügung:

<code>_SMTPFrom</code>	Absenderadresse und -name.
<code>_SMTPTo</code>	Empfängeradresse und -name.
<code>_SMTPCc</code>	Empfängeradresse und -name für Kopie.
<code>_SMTPBcc</code>	Empfängeradresse und -name für unsichtbare Kopie.
<code>_SMTPSubject</code>	Nachrichtenbetreff.
<code>_SMTPReplyTo</code>	Antwortadresse und -name.
<code>_SMTPPriority</code>	Priorität (maximal 1, mindestens 5, Standard: 3)

Mit den Datentypen `_SMTPTo`, `_SMTPCc` und `_SMTPBcc` lassen sich mehrere Empfänger definieren.

Die folgende Datentypen stehen für den E-Mail-Inhalt zur Verfügung:

<code>_MailLine</code>	Zeile.
<code>_MailBuffer</code>	Textpuffer.
<code>_MailFile</code>	Dateipfad und -typ.

Die Datentypen `_MailLine` und `_MailBuffer` enthalten den E-Mail-Text und lassen sich mit der Option `_MimeTextHTML` kombinieren, um formatierten HTML-Text zu definieren. Mit dem Datentyp `_MailFile` kann der Entwickler der E-Mail Anhänge hinzufügen.

```
tMail->MailData(_MailLine | _MimeTextHTML, '</html>');
```

Bei Bedarf kann die E-Mail auch um Dateianhänge erweitert werden:

```
tMail->MailData(_MailFile | _MimeApp,
    'C:\ProgramData\MyApplication\Log.txt');
```

Zum Schluß wird das Mailobjekt geschlossen und die Nachricht versandt. Dafür genügt der Aufruf der Funktion *MailClose*():

```
tMail->MailClose(_SMTPSendNow);
```

Um dem Benutzer einer Konzept-16-Anwendung die manuelle Eingabe von formatiertem Text zu ermöglichen, bietet sich das RTF-Objekt an. Damit kann der Text nicht nur eingegeben, sondern auch mit Formatierungen wie fett, kursiv oder unterstrichen ausgezeichnet werden. Außerdem können damit die Schriftart, -farbe, -größe und -ausrichtung angepaßt werden. Den eingegebenen und formatierten Text gilt es dann nur noch, mit dem Mail-Objekt in

MailClose

MailClose versendet beziehungsweise verwirft die E-Mail und schließt das Mail-Objekt.

```
MailClose(
    MailHdl      : handle; // Mail-Objekt
    Process      : int;    // Verarbeitungsart
) : int          // Resultat
```

Die Verarbeitungsart `_SMTPSendNow` versendet die E-Mail, `_SMTPDiscard` verwirft die Nachricht.

eine E-Mail zu verpacken. Mit der Funktion *Win.RTF-SaveHTML()* - zu finden auf der DVD zu dieser Ausgabe - wird der Inhalt eines RTF-Objekts als HTML-Text in einen Textpuffer gespeichert.

```
// ++++++
// + RTF-Text als HTML-Text speichern +
// ++++++

sub Win.RTFsaveHTML(
    aWinRTF      : handle; // RTFEdit-Objekt
    aText        : handle; // Textpuffer )
{ } //...
```

Die Funktion *Win.RTFsaveHTML()* ermittelt die Formatierungen eines RTF-Objekts und überträgt sie in HTML in einen Textpuffer. Dieser Textpuffer kann anschließend über ein Mail-Objekt einer E-Mail hinzugefügt werden:

```
// Textpuffer für ASCII-Text anlegen
tTextPlain # TextOpen(32);
// Textpuffer für HTML-Text anlegen
tTextHTML # TextOpen(32);

// RTF-Text als ASCII-Text speichern
tWinRTFEdit->WinRTFSave(_WinStreamBufText, _WinRTFSaveASCII,
    tTextPlain);
// RTF-Text als HTML-Text speichern
tWinRTFEdit->Win.RTFsaveHTML(tTextHTML);
// ASCII-Text zu E-Mail hinzufügen
tMail->MailData(_MailBuffer, CnvAI(tTextPlain,
    _FmtInternal));
// HTML-Text zu E-Mail hinzufügen
tMail->MailData(_MailBuffer |
    _MimeTextHTML, CnvAI(tTextHTML,
    _FmtInternal));

// Textpuffer für ASCII-Text schließen
tTextPlain->TextClose();
// Textpuffer für HTML-Text schließen
tTextHTML->TextClose();
```

Mit dem Mail-Objekt besteht in jeder Situation die Möglichkeit, E-Mails zu verschicken. Seien es automatisch generierte für das Überwachen eigenständiger Prozesse oder manuell verfaßte E-Mails – mit dem Mail-Objekt versorgen Konzept-16-Applikationen ihre Benutzer jederzeit mit wichtigen und aktuellen Informationen. ■