TSMTP Component

Properties Methods Events Tasks

TSMTP is a component for Borland Delphi, which allows to send Internet mail. If you have PPP or SLIP connection to the Internet, you can use this component in your applications. It fully supports Simple Mail Transfer Protocol (SMTP, rfc 821). You can attach files to the text messages and encode attachments using such popular methods as UUEncoding and MIME (rfc 1521). These features are built in. Encoding methods are fully compatible with the following mail retrieval agents: AOL, Eudora, Microsoft Exchange, Pegasus, Pine and probably more, we performed extensive tests only with these agents.

Properties

Attachments
Body
CC
Encoding
From
Recipient
Server
Subject
TimeOut

Methods

Cancel
Connect
Disconnect
SendBody
SendEnvelope
SendHeaders
Send
WriteLogFile

Events

OnStatusChange

Using the TSMTP component

See Also TSMTP Reference

Use TSMTP component to send Internet mail

if you want to send just one single mail message, you have to set appropriate properties and call the following sequence of methods:

```
with SMTP1 do
try
  Connect;
  Send;
finally
  Disconnect;
end;
```

if you want to send a bunch of messages then:

```
with SMTP1 do
try
  Connect;
  for i:=1 to NumOfMessages do
  begin
{Set mail attributes here}
    Recipient:=...;
    Body:=...;
    .
    .
    Send;
  end;
finally
  Disconnect;
end;
```

for more information have a look at smtpmail.dpr project which is included with this package.

Limitations
Installation

Limitations

There are some limitations related to the sizes of attachments and messages. First of all, this version of TSMTP does not support multiple attachments, but it will be available soon. There is a limit in TList objects of MaxInt div 2 = 16,383, so number of encoded lines can not exceed this limit. It means size of original files can be up to 700Kb for UUEncoding and 900KB for MIME. In the future releases this problem will be fixed. There exists also a limit about the sizes of messages in the SMTP servers.

Installation

There are two ways to use TSMTP. **First way** is standard one - install it into your Component Palette. To do it, copy the following files:

winsock.pas
winsock.inc
errno.res
mime.pas
uucode.pas
smtp.dcu (for registered version - smtp.pas)
smtp.dcr

into your *delphi\lib* directory, click *options/install* components, then *add*, *browse* and select the file smtp.dcu (for unregistered version) or smtp.pas (for registered version), then click *OK* and TSMTP component will appear in the *Internet* tab of your component palette.

But there can be a problem if you already *have* installed another components, which are using winsock. You can get *duplicate resource identifier error*. There are two solutions. First: uninstall existing winsock related components and install TSMTP, Second: Use the second way of using TSMTP.

Second way is: Don't install TSMTP into your component palette. Copy all files containing in the smtpmail.zip file into separate directory and use TSMTP as class, i.e. create it at run time, as it is done in the attached sample application.

Registration

Registered users will receive the full source code of TSMTP and the right of unlimited usage of this component, technical support via Email, all minor upgrades and bug fixes (if there are any). They also will be notified about all further enhancements and about the release of TPOP3 component, which will allow you to receive Internet mail.

CompuServe Users:

TSMTP can be registered on CompuServe. This is the cheapest way to register. Just GO SWREG, search for registration number 10338, and follow the instructions. Registration fee is just US$14.95. Registered version, including full source code will be sent via CompuServe mail.

Credit Card Orders Only:

Credit card orders can be made from Public (software) Library, but it will cost you $16.95 plus $1.00 for shipping and handling to USA and Canada and $2.00 to other countries.

You can order with MC, Visa, Amex, or Discover from PsL by calling 800-2424-PsL or 713-524-6394 or by FAX to 713-524-6398 or by CIS Email to 71355,470. You can also mail credit card orders to PsL at P.O.Box 35705, Houston, TX 77235-5705.

Please refer to product #14510 when ordering from PsL.

THE ABOVE NUMBERS ARE FOR ORDERS ONLY.

Any questions about the status of the shipment of the order, refunds, registration options, product details, technical support, volume discounts, dealer pricing, site licenses, etc, must be directed to the addresses listed in the About topic of this document.

To insure that you get the latest version, PsL will notify us the day of your order and we will ship the product directly to you.

By regular mail

To order by regular mail print out the file regform.wri, fill it in and send along with your payment. Registered version of TSMTP will be forwarded to you via Internet, CompuServe or AOL mail, FTP or by regular mail. See REGFORM.WRI for more information.

Disclaimer

This Software and the accompanying files are provided "As Is" and without warranties as to performance of merchantability or any other warranties whether expressed or implied. No warranty of fitness for a particular purpose is offered. Any liability of the seller will be limited exclusively to product replacement or refund of purchase price.

About

TSMTP component for Delphi, v. 1.1.

compuserve: 75231,330
internet: 75231.330@compuserve.com
www: http://www.icacomp.com/customers/agogava

Postal Address:
ArGo Software Design
200 Balliol Street, #1405
Toronto Ontario M4S 1C6
Canada.

Attachments Property

Example

Applies to:

TSMTP Component.

Declaration:

```
property Attachments : TStrings;
```

Files listed in this property will be encoded and attached to the email message. If any of the files does not exist, or at least one file is too large (see limitations) exception will be raised and user will be prompted that attachment is not valid and if he wants to proceed with sending mail without attachments.

This property has been changed from the previous version.

Example

```
OpenDialog1.Options:=[ofAllowMultiSelect];
if OpenDialog1.Execute then
  TSMTP.Attachments:=OpenDialog1.Files;
```

Body Property

Example

Applies to:
TSMTP Component.

Declaration:

```
property Body : TStrings;
```

This is a body of mail message by itself. You have to use ':=' operation to assign a value to this property.

Example

```
SMTP1.Body:=Memo1.Lines;
```

CC Property

Example

Applies to

TSMTP component.

Declaration

```
property CC : TStrings;
```

Carbon Copy - i.e. list of addresses and names of additional recipients. Syntax of these lines is the same as for From property.

Encoding Property

applies to
TSMTP component

declaration
```
Encoding : TEncoding;
```

Encoding method of attachment. if etUU then attachment will be UUEncoded, if etMIME then - MIME will be used. Default is etMIME.

Example

```
with TSMTP.CC do
begin
  Add('jdoe@microsoft.com|John Doe');
  Add('jsmith@somenet.net|James Smith');
  Add('tadams@torfree.net');
end;
```

TEncoding Type

Unit
SMTP

Declaration
```
TEncoding = (etUU, etMIME);
```

Description
a type of Encoding property of the TSMTP component. Indicates the method of encoding of the binary attachments to mail messages.

From Property

Example

applies to

TSMTP component

declaration

```
property From : string;
```

description

Email address and name of sender. You can enter both email address and name of sender in this single string. Just put | (pipe) symbol between them. If the string consists of only one part, i.e. there is no pipe (|) symbol there, all string will be considered as Internet address. The same rules apply to Recipient and CC properties.

Example

```
SMTP1.From:='jdoe@somedomain.com|John Doe';
```

Recipient Property

Example

applies to

TSMTP component

Declaration

```
property Recipient : string;
```

description

The email address and the name of recipient. Format is the same as for From property.

Example

```
TSMTP1.Recipient:='bcarter@inetcom.edu|Bill Carter';
```

Server Property

Example

applies to

TSMTP component

declaration

```
property Server : string;
```

desription

Address of SMTP server. Your application will make an attempt to connect to this server to send the mail. Can have format *00.00.00* or *mail.somehost.com*. In second case TSMTP attempts to resolve the host name using Winsock *gethostbyname()* function.

Example

```
SMTP1.Server:='mail.compuserve.com';
```

or

```
SMTP1.Server:='179.231.11.1';
```

In first case will be made an attempt to resolve the host name, in second case the address will be converted directly to IP.

Subject Property

Example

applies to

TSMTP component

declaration

```
property Subject : string;
```

description

A subject line of mail message.

Example

```
TSMTP1.Subject:='Information you have requested';
```

TimeOut Property

Example

Applies to

TSMTP component

declaration

```
property TimeOut : Integer;
```

description

Each blocking operation in TSMTP component is checked for timeout. For example, if gethostbyname() function does not return for TimeOut seconds, exception will be raised and task will be terminated. Default value is 20.

Example

```
TSMTP1.TimeOut:=30;
```

sets timeout to 30 seconds.

Cancel Method

Example

applies to

TSMTP component

declaration

```
procedure Cancel;
```

description

Cancels current operation. If connection is in progress, tries to disconnect SMTP server sending QUIT command and closes open sockets if there is one.

Example

```
function CancelButtonClick(Sender : TObject);
begin
  SMTP1.Cancel;
end;
```

Connect Method

Example

Applies to

TSMTP component

declaration

```
procedure Connect;
```

description

Connect method performs the following tasks: First of all in encodes Attachment if there is any and stores it in the UULines object, then checks for Server address and tries to resolve remote address, then tries to get SMTP port number, creates a socket and opens it and tries to make connection with remote SMTP server. If any of these operations was not successful, exception will be raised. Only exception from this rule is the step of getting SMTP port number. If this operation is not successful, well known port number (25) will be used and no exception will be raised.

Example

```
SMTP1.Connect;
```

Disconnect Method

Example

applies to

TSMTP component

declaration

```
procedure Disconnect;
```

description

If connection has not been established this procedure just closes open socket, if there is one, otherwise it sends QUIT command to the server. In one word, it disconnects the socket with remote server.

Example

```
SMTP1.Disconnect;
```

SendBody Method

Example

applies to

TSMTP component

declaration

```
procedure SendBody;
```

description

Sends a message Body and Attachment.

Example

```
SMTP1.SendBody;
```

for more information regarding usage of this method, see Usage of TSMTP component.

Send Method

Applies to
TSMTP component

declaration
```
procedure Send;
```

description
Send method sends a single message. Here is its full source code:

```
procedure TSMTP.Send;
begin
  SendEnvelope;
  SendHeaders;
  SendBody;
end;
```

SendEnvelope Method

Applies to
TSMTP component

declaration
```
procedure SendEnvelope;
```

description
This method sends envelope information, i.e. commands to the server. Envelope information contains data about sender, recipients etc.

SendHeaders Method

Applies to

TSMTP component

declaration

```
procedure SendHeaders; virtual;
```

description

This procedure sends message headers. It includes the following fields: From, To, Subject, if there are MIME attachements, then MIME-Version, and other MIME related fields (see RFC 1521), and some fields which are introdused by ArGo Software Design. Their names start with X- (according to RFC 822). These fields will appear only if the message contains binary attachments. They are implemented for future use in TPOP3 component. Method is declared as virtual so you can modify it and send your own headers.

WriteLogFile Method

Applies to

TSMTP component

declaration

```
procedure WriteLogFile;
```

description

This method is provided for debug purposes. It will work only if the field of TSMTP.LogFileName field is not blank. You can call this method after completion of your task to write log file and find out more about the conversation between your application and SMTP server.

OnStatusChange Event

Example

Applies to

TSMTP component

declaration

```
property OnStatusChange : TNotifyEvent;
```

description

This event allows you to check the status of your application. You can assign this event to your SMTP component instance and trigger the current status using Status property of TSMTP component. This property does not appear in object inspector, because it is not declared as published, but it is accessible because it is declared as public. Type of Status property is TStatus.

TStatus Type

Unit
SMTP

declaration

```
TStatus = (msIdle,msResolving,msConnecting,msHeaders,
      msEnvelope,msBody,msAttachment,msDisconnecting,msError,
      msCancel,msEnCode);
```

All values are self explanatory.

Example

```
procedure TForm1.SMTP1StatusChange(Sender : TObject);
var
  s : string;
begin
  case SMTP1.Status of
   msIdle : s:='';
   msResolving : s:='Resolving remote host';
   msConnecting : s:='Connecting to server';
   msHeaders : s:='Sending headers';
   msEnvelope : s:='Sending commands';
   msBody : s:='Sending message body';
   msAttachment : s:='Sending attachment';
   msDisconnecting : s:='Disconnecting';
   msError : s:='Error';
   msCancel : s:='Canceled';
   msEnCode : s:='Encoding the attachment';
  end;
  StatusBar.Caption:=s;
end;
```

Installing this help file

You can use this help file separately, or merge its index to the Delphi master Help index file.

To merge your keyword file into the Delphi master Help index,

1 Make sure you have placed the file SMTP.KWF along with the SMTP.HLP file in the directory with the compiled unit that contains your TSMTP component, which is SMTP.DCU.

2 Run the HELPINST application. HELPINST is a Windows application installed with Delphi.

3. Within HELPINST open the file \DELPHI\BIN\DELPHI.HDX.

4. Click Add Keywod file and select SMTP.KWF. Help file for TSMTP will be merged with Delphi help.

Packaging

The following files are included in the unregistered version of TSMTP component:

smtp.dcu	- compiled unit, containing TSMTP component;
smtp.dcr	- component resource file;
winsock.pas	- winsock interface;
winsock.inc	- include file used by winsock.pas (declarations of types and constants);
errno.res	- resource file containing error strings for winsock;
mime.pas	- unit handling MIME encoding of binary attachments;
uucode.pas	- unit handling UU encoding of binary attachments;

The following files build the sample application smtpmail.exe, which is fully functional SMTP mail agent:

smtpmail.dpr
mmail.pas
mmail.dfm
smtpsu.pas
smptsu.dfm

And all the documentation is in the following files:

readme.wri	- documentation in windows write format;
file_id.diz	- standard BBS file description;
regform.wri	- registration order form;
smtp.hlp	- this file;
smtp.kwf	- keyword file for merging to Delphi master Help Index.

Changes from the Previous Version

Version 1.1 has some additional features. Major one is - support of multiple binary attachments. Because of this the old *Attachment* property has been changed to Attachments, which is now declared as TStrings.

There was a bug in TSMTP version 1.0: it was impossible to send different messages connectiong only once to the server. It is fixed now.

All the following applies to you if you are using MIME for encoding the attachments.

TSMTP now supports 2 kinds of MIME: Base64 and Quoted-Printable. **Base64** is good for "real" binary files, e.g. zipped archives, jpg and gif pictures and so on, but it increases the size of the original file by 25%. **Quoted-Printable** encoding is ideal for ascii files, where can be small amount of characters, which cannot be transmitted via Internet mail, for example, characters with the foreign letters, such as Ç, ü and so on, which occupy upper portion of the ASCII table. In this case only foreign characters will be encoded, so size of the original file will stay approximately the same. Qouted-Printable encoding also handles long lines. The problem is, Internet mail allows only 512 characters per line, but Pascal has more strict limitation: 255 characters per line. Quoted-Printable encoding handles this problem, it breaks the lines.

TSMTP looks for the extension of the attachment and uses Quoted-Printable encoding *only* if the first five characters of the name of MIME type of the attachment is text/. In this implementation of TSMTP there are defined three MIME types of this kind:

```
text/plain        extension *.txt
text/rtf          extension *.rtf
text/html         extension *.htm
```

The procedure *GetContentType*, located in the *mime.pas* unit handles this. You can modify this part and include another MIME types, if you wish.