w3 JMail

Object reference

The worlds most popular e-mail component!

JMail has to today's date been downloaded over 150 000 times and is globally spread on servers. JMail is a small e-mail component based on ASP tecnology and is suitable for e-mailing forms from webpages without need of an e-mail address.

Dimac AB
Garnisonsgatan 25
Box 222 28
250 24 Helsingborg

Support: http://support.dimac.net
Support e-mail: support@dimac.net
Homesite: http://www.dimac.net
E-mail: info@dimac.net

Properties

Count : *Integer*

Returns the number of messages on the pop3 server.

```
'i = pop3.Count'
```

Log : *String*

This is the log created by JMail when loging is set to true

```
'Response.Write( POP3.Log )'
```

Logging : *Boolean*

Enables/Disables logging in JMail. Default value is false.

```
'POP3.Logging = true'
```

Messages : *Pointer*

Returns the Messages object through this you can access the messages.

```
'set messages = POP3.Messages'
```

Size : *Integer*

Returns the total size of you mailbox.

```
'size = POP3.Size'
```

Methods

Connect(*Username, Password, Server, Port* **)**

Opens the connection to the server. The Port argument is optional and defaults to 110.

```
'POP3.Connect "john", "qwerty", "mail.myDomain.com"'
```

DeleteMessages( )

Deletes all messages form the mail server.

```
'Mailbox.DeleteMessages'
```

DeleteSingleMessage(*MessageID* **)**

Deletes a single message on the mailserver.

```
'Mailbox.DeleteSingleMessage 1'
```

Disconnect( )

Closes the connection to the server.

```
'POP3.Disconnect'
```

DownloadHeaders( )

Downloads all headers and adds them in the messages collection.

```
'Mailbox.DownloadHeaders'
```

DownloadMessages( )

Downloads all messages.

```
'Mailbox.DownloadMessages'
```

DownloadSingleHeader(*MessageID* **)**

Downloads the headers of a single message, and adds them to the messages collection.

```
'Mailbox.DownloadSingleHeader 1'
```

DownloadUnreadMessages( )

Downloads all unread (messages that has not been downloaded by ANY clientsoftware) mails. The mails are added to the messages collection. Note this command has been removed from the POP3 specification (RFC1725), so it may not be supported by all mailservers.

```
'Mailbox.DownloadUnreadMessages'
```

GetLastUnreadMessage( ) : Integer

Returns the ID of the first unread (message that has not been downloaded by ANY clientsoftware) message. Return 0 if no messages has been accessed, -1 if this command is not supported by the server. Note this command has been removed from the POP3 specification (RFC1725), so it may not be supported by all mailservers.

```
'lastMessage = Mailbox.GetLastUnreadMessage'
```

GetMessageUID(*MessageID* **)** : String

Returns the server´s unique id for this message.

```
'Mailbox.GetMessageUID 1'
```

Properties

Count : *Integer*

Returns the number of messages in the collection.

```
'i = Messages.Count'
```

Item : *Pointer*

Returns a message object.

```
'set msg = Messages.Item(0)'
```

Methods

Clear( )

Clears the collection. Note it will NOT remove ANY mails from your mailserver.

```
'Messages.Clear'
```

w3 JMail

Properties

About : *String*

Some useful information.

```
'Response.Write( Message.About )'
```

Attachments : *Pointer*

Returns the Attachments collection.

```
'set attachments = Message.Attachments'
```

Body : *String*

Returns the Message\ Body.

```
'Response.Write( Message.Body )'
```

BodyText : *String*

Returns the entire raw unparsed body (Text - Headers.Text).

```
'Response.Write( Message.BodyText )'
```

Charset : *String*

This is the charset of the message. The default is "US-ASCII"

```
'JMail.Charset = "US-ASCII"'
```

ContentTransferEncoding : *String*

Specifies the content transfer encoding. The default is "Quoted-Printable"

```
'JMail.ContentTransferEncoding = "base64"'
```

ContentType : *String*

Returns the Bodie\ Content-Type.

```
'Response.Write( Message.ContentType )'
```

Date : *Date*

Returns the DateTime when the message was sent.

```
'Response.Write( Message.Date )'
```

DeferredDelivery : *Date*

Sets defered delivery of messages. If the mailserver supports it the message wont be delivered before this date and time.

```
'JMail.DeferredDelivery = new Date( 2000, 02, 17 ).getVarDate();'
```

Encoding : *String*

This can be used to change the default Attachment encoding from base64. Valid options are "base64", "uuencode" or "quoted-printable"

```
'JMail.Encoding = "base64"'
```

EncryptAttachments : *Boolean*

Set to true all attachments will be encrypted too if encryption is enabled.The default value is TRUE.

```
'JMail.EncryptAttachments = true'
```

ErrorCode : *Integer*

Contains the error code if JMail.silent is set to true

```
'Response.Write( JMail.ErrorCode );'
```

ErrorMessage : *String*

Contains the error message if JMail.silent is set to true

```
'Response.Write( JMail.ErrorMessage );'
```

ErrorSource : *String*

Contains the error source if JMail.silent is set to true

```
'Response.Write( JMail.ErrorSource );'
```

From : *String*

Returns the senders EMail.

```
'Response.Write( Message.From )'
```

FromName : *String*

Returns the senders Name.

```
'Response.Write( Message.FromName )'
```

Headers : *Pointer*

Returns the Headers object.

```
'set Headers = Message.Headers'
```

ISOEncodeHeaders : *Boolean*

Encodes header stings according to iso-8859-1 character sets. The default is true.

```
'JMail.ISOEncodeHeaders = false'
```

Log : *String*

This is the log created by JMail when loging is set to true

```
'Response.Write( JMail.Log );'
```

Logging : *Boolean*

Enables/Disables logging in JMail

```
'JMail.Logging = true'
```

MailData : *String*

The raw maildata as the mail would like like when it is delivered.

```
'Response.write( JMail.MailData )'
```

MailDomain : *String*

This can be used to override the EHLO/HELO statement to your mailserver

```
'JMail.Maildomain = "hello.world.com"'
```

MailServerPassWord : *String*

The password for SMTP server authentication.

```
'JMail.MailServerPassWord = "mypassword"'
```

MailServerUserName : *String*

Used to specify the username for SMTP server authentication if the mailserver requires a user to log in.

```
'JMail.MailServerUserName = "myUserName"'
```

MessageOrder : *String*

If the body has multiple parts, you can set this to which body content type you want. Default value is "text/plain;text/html"

```
'Message.MessageOrder = "text/html;text/plain"'
```

MimeVersion : *String*

Specifies the mime version. The default is "1.0"

```
'JMail.MimeVersion = "1.0"'
```

MsPickupdirectory : *String*

The path to the pickup directory of MS SMTP service.If not set, JMail will autodetect the path from registry settings.

```
'JMail.MsPickupdirectory = "c:\myfolder"'
```

PGPEncrypt : *Boolean*

Set to true, the mail will be encrypted when the message is sent, using PGP.

```
'JMail.PGPEncrypt = true'
```

PGPPassphrase : *String*

The PGP passphrase used when signing.

```
'JMail.PGPPassPhrase = true'
```

PGPSign : *Boolean*

Set to true, the mail will be signed when the message is sent, using PGP.

```
'JMail.PGPSign = true'
```

PGPSignkey : *String*

A email address or a key id identifying the key to be used for signing.

```
'JMail.PGPSingKey = "charlie@hisdomain.com"'
```

Priority : *Byte*

Returns the Message\ priority. 3 is normal priority

```
'Response.Write( Message.Priority )'
```

Recipients : *Pointer*

Returns the Recipients collection.

```
'set recipients = Message.Recipients'
```

RecipientsString : *String*

Readonly property of all recipients of this message

```
'Response.Write( JMail.Recipients )'
```

ReplyTo : *String*

Specifies a optional reply address

```
'JMail.ReplyTo = "president@dimac.net"'
```

ReturnReceipt : *Boolean*

Specifies wether or not the sender requires a return receipt. The default value of the property is "false"

```
'JMail.ReturnReceipt = true'
```

Silent : *Boolean*

Set to true, JMail will not trow exceptions. Instead JMail.execute() will return true or false depending on the success of the operation

```
'JMail.silent = true'
```

SimpleLayout : *Boolean*

Set to true to reduce the number of headers JMail produces.

```
'JMail.SimpleLayout = true'
```

Size : *Integer*

Returns the total size of the message in bytes.

```
'Response.Write( Message.Size )'
```

Subject : *String*

Returns the Message\ Subject.

```
'Response.Write( Message.Subject )'
```

Text : *String*

Returns the entire message source.

```
'Response.Write( Message.Text )'
```

UsePipelining : *Boolean*

Overrides if JMail should use pipelining on a server that supports it.

```
'JMail.Pipelining = false'
```

Version : *String*

Return version information.

```
'Response.Write( Message.Version )'
```

Methods

AddAttachment(*FileName, ContentType* **)** : String

Adds a file attachment to the mssage

```
'JMail.AddAttachment "c:\autoexec.bat"'
```

AddCustomAttachment(*FileName, Data* **)** : String

Adds a custom attachment. This can be used to attach "virtual files" like a generated text string or certificate etc.

```
'JMail.AddCustomAttachment "readme.txt", "Contents of file"'
```

AddHeader(*XHeader, Value* **)**

Adds a user defined X-header to the message

```
'JMail.AddHeader "Originating-IP","193.15.14.623"'
```

AddNativeHeader(*Header, Value* **)**

Adds a header to the message

```
'JMail.AddNativeHeader "MTA-Settings", "route"'
```

AddRecipient(*emailAddress, recipientName, PGPKey* **)**

Adds a recipient to the message

```
'JMail.AddRecipient "info@dimac.net"'
```

AddRecipientBCC(*emailAddress, PGPKey* **)**

Adds a Blind Carbon Copy recipient to the message

```
'JMail.AddRecipientBCC "someone@somedomain.net"'
```

AddRecipientCC(*emailAddress, recipientName, PGPKey* **)**

Adds a Carbon Copy recipient to the message

```
'JMail.AddRecipientCC "someone@somedomain.net"'
```

AddURLAttachment(*bstrURL, bstrAttachAs, bstrAuth* **)** : String

Downloads and adds an attachment based on an URL. A seconds argument, "AttachAs", is used for specifying the filename that the attachment will receive in the message. A third and optional argument is used for optional WWW-Authentication.

```
'JMail.AddURLAttachment "http://download.dimac.net/jmail/jmail.exe", "jmail.exe"'
```

AppendBodyFromFile(*FileName* **)**

Appends body text from a file

```
'JMail.AppendBodyFromFile "c:\mytext.txt"'
```

AppendText(*Text* **)**

Append "text" to body

```
'JMail.AppendText "Text appended to message Body"'
```

Clear( )

Clears the message object, and gives you a new clean message.

```
'Message.Clear'
```

ClearAttachments( )

Clears the list of attachments

```
'JMail.ClearAttachments'
```

ClearCustomHeaders( )

Clears all custom headers

```
'JMail.ClearCustomHeaders();'
```

ClearRecipients( )

Clear the recipient list

```
'JMail.ClearRecipients'
```

Close( )

Force JMail to close an cached connection to a mailserver.

```
'JMail.Close();'
```

DecodeHeader(*Header* **)** : String

Decodes a message header.

```
'Response.Write( Message.DecodeHeader( Headers["ReplyTo"] ) )'
```

ExtractEmailAddressesFromURL(*bstrURL, bstrAuth* **)**

Downloads and adds email addresses from an URL.

```
'JMail.ExtractEmailAddressesFromURL "http://duplo.org/generateEmailList.asp"'
```

GetMessageBodyFromURL(*bstrURL, bstrAuth* **)**

Clears the body of the message and replaces it with the contents of the URL. The contenttype is automaticly set to match the contentype of the URL. The second argument (login and password) is optional

```
'JMail.GetMessageBodyFromURL( "http://duplo.org/", "login:password" )'
```

KeyInformation(*keyIdentifier* **)** : Pointer

Returns an IPGPKeys object holding information for the keys matching the supplied identifier.

```
' keys = JMail.KeyInformation("charlie@hisdomain.com")'
```

LoadFromStream(*Stream* **)**

Loads a message from a stream. Note the stream data must be compatible with the message format described in RFC822.

```
'Message.LoadFromStream( myStream )'
```

LogCustomMessage(*Message* **)**

Logs a custom user message to the JMail log. This function works ONLY if loging is set to true

```
'JMail.LogCustomMessage( "Hello world" );'
```

nq( )

Append the mail to mail-queue and return.

```
'Message.nq'
```

ParseMessage(*MessageSource* **)**

Parses a message. MessageSource must be compatible with the message format described in RFC822.

```
'Message.ParseMessage myHeaders & vbCrLf & myBody'
```

SaveToStream(*Stream* **)**

Saves the message source (RFC822 compatible message) to a stream.

```
'Message.SaveToStream( myStream )'
```

Send(*mailServer, enque* **)**

Sends the message.Mailservers is a string with one or more hostnames separeted by a semicolon.A username and password can also be provided for each server in the format username:password@myhost.mydomain.com.

```
'Message.Send( "myMailServer" ),
Message.Send("myUserName:mypassword@mymailserver.mydomain.com")'
```

SendToNewsGroup(*ServerName, Newsgroups* **)**

Sends the message to Newsgroups (Seperated by a ",") using the NNTP server specified.

```
'SendToNewsGroup( myNNTPServer, "alt.test, alt.test.test" )'
```

VerifyKeys(*keyString* **)** : Boolean

Returns true if all the supplied keys where found in the local keyring.

```
'JMail.VerifyKeys "recipient1@hisdomain.com,recipient2@hisdomain.com"'
```

w3 JMail

JMail.Headers
- Text : *String*
- GetHeader(*Headername*) : String

Properties

Text : *String*

Returns All headers.

```
'Response.Write( Headers.Text )'
```

Methods

GetHeader(*Headername* **)** : String

Returns the value of HeaderName.

```
'Response.Write( Headers.GetHeader( "X-Mailer" ) )'
```

w3 JMail

- JMail.Recipients
  - Count : *Integer*
  - Item : *Pointer*
  - Add(*Value*)
  - Clear( )

Properties

Count : *Integer*

Returns the number of recipients in the collection.

```
'i = Recipients.Count'
```

Item : *Pointer*

Returns a recipient object.

```
'set re = Recipients.Item(0)'
```

Methods

Add(*Value* **)**

Adds a Recipients to the collection.

```
'Recipients.Add( re )'
```

Clear( )

Clears the Collection.

```
'Recipients.Clear'
```

w3 JMail

- JMail.Recipient
  - EMail : *String*
  - Name : *String*
  - ReType : *Integer*
  - New(*Name, EMail, recipientType*) : Pointer

Properties

EMail : *String*

Returns the recipient's EMail.

```
'Response.Write( Recipient.EMail )'
```

Name : *String*

Returns the recispient's name.

```
'Response.Write( Recipient.Name )'
```

ReType : *Integer*

Returns the recipient's type (To = 0, CC = 1, BCC = 2).

```
'Response.Write( Recipient.ReType )'
```

Methods

New(*Name, EMail, recipientType* **)** : Pointer

Creates a new Recipient, whom you can add to the Recipients collection.

```
'set re = Recipient.New( "Firstname Lastname", "name@domain.com", 0 )'
```

w3 JMail

- JMail.Attachments
  - Count : *Integer*
  - Item : *Pointer*
  - Add(*Attachment*)
  - Clear( )

Properties

Count : *Integer*

Returns the number of Attachments in the collection.

```
'i = Attachments.Count'
```

Item : *Pointer*

Returns a attachment object.

```
'set attachment = Attachments.Item(0)'
```

Methods

Add(*Attachment*)

Adds a Attachments to the collection.

```
'Attachments.Add( re )'
```

Clear( )

Clears the collection.

```
'Attachments.Clear'
```

Properties

ContentType : *String*

Returns the attachment's ContentType.

```
'Response.Write( Attachment.ContentType )'
```

Data : *String*

Returns the Attachment's data.

```
'Response.Write( Attachment.Data )'
```

Name : *String*

Returns the attachment's filename.

```
'Response.Write( Attachment.Name )'
```

Size : *Integer*

Returns the attachment's Size.

```
'Response.Write( Attachment.Size )'
```

Methods

New(*FileName, ContentType, Data* **)** : Pointer

Creates a new attachment which you can add to the Attachments collection. If Data is specified JMail creates a custom attachment containing the data, else it reads FileName from disk.

```
'set attachment = Attachment.New( "myAttachment.text", "text/plain", "this is my new
text file" )'
```

SaveToFile(*FileName* **)**

Saves the attachment to disk.

```
'SaveToFile "c:\incommingAttachments\" + Attachment.Name'
```

Properties

Item : *String*

Set your merging variables manually. Note you can not combine this with recordset merges.

```
'MailMerge.Item( "CustomerName" ) = "Lisa Nilsson"'
```

MailTemplate : *Pointer*

Set your own created Message object (it will serve as a template in the merge proccess).

```
'MailMerge.MailTemplate = myMsg'
```

MergeAttachments : *Boolean*

Do you want to have attachments merged in the same manor as body and headers? Default value is False.

```
'MailMerge.MergeAttachments = True'
```

Methods

BulkMerge(*RecordSet, enque, Maildestination* **)**

Merge entire Recordset with MailTemplate and sends Enques it in MailDestination or sends it (if Enque = False) to the Mailserver specified in Maildestination.

```
'MailMerge.BulkMerge( myRS, true, "c:\inetpub\mailroot\pickup" )'
```

Expand( ) : Pointer

Merges MailTemplate with user defined variables specified in the Item property.

```
'MailMerge.Expand'
```

ExpandFromRecordSet(*RecordSet* **)** : Pointer

Merges one row from an ADO Recordset with MailTemplate.

```
'set msg = MailMerge.ExpandFromRecordSet( myRS )'
```

SetDebugMode(*TestMailAddress, TestCount* **)**

Tells Mailmerge to enter debugging mode. All recipients in your mails will be set to TestMailAddress, TestCount mails will be sent to you.

```
'MailMerge.SetDebugMode( "myEMail@company.com", 10 )'
```

JMail.PGPKeys
- Count : *Integer*
- Item : *Pointer*

Properties

Count : *Integer*

The number of keys in collection

```
'Response.write( keys.Count )'
```

Item : *Pointer*

Returns PGPKeyInfo objects from the collection.

```
' key = keys.Item(0)'
```

JMail.PGPKeyInfo
- KeyCreationDate : *String*
- KeyID : *String*
- KeyUser : *String*

Properties

KeyCreationDate : *String*

The date the key was created.

`null`

KeyID : *String*

The id of the key.

`null`

KeyUser : *String*

The name of the user who created the key.

`null`

w3 JMail

- JMail.SpeedMailer
  - EnqueMail(*FromEMail, RecipientEMails, Subject, Body, MsPickupdirectory*)
  - SendMail(*FromEMail, RecipientEMails, Subject, Body, MailServers*)

Methods

EnqueMail(*FromEMail, RecipientEMails, Subject, Body, MsPickupdirectory*)

Writes the mail to the mailqueue and returns. All data is provided through parameters.

```
'JMail.EnqueMail("me@mydomain.com","recipient@hisdomain.com","This is a test","Example")'
```

SendMail(*FromEMail, RecipientEMails, Subject, Body, MailServers*)

Send a mail with SMTP, all maildata is provided through parameters.

```
'JMail.SendMail("me@mydomain.com","recipient@hisdomain.com","This is a test","Example","mail.mydomain.com")'
```