|  |
| --- |
| [**TOC**](#gjdgxs) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | The README file | M.T. Rose | |  | Dover Beach Consulting, Inc. | |  | February 22, 2000 | |

**Tcl MIME**

### **Abstract**

Tcl MIME generates and parses MIME body parts.

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **Table of Contents**

[**1.**](#30j0zll)**SYNOPSIS**  
 [**1.1**](#1fob9te)**Requirements**  
 [**1.2**](#3znysh7)**Copyrights**  
 [**2.**](#2et92p0)**SYNTAX**  
 [**3.**](#tyjcwt)**SEMANTICS**  
 [**3.1**](#3dy6vkm)**mime::initialize**  
 [**3.2**](#1t3h5sf)**mime::finalize**  
 [**3.3**](#4d34og8)**mime::getproperty**  
 [**3.4**](#2s8eyo1)**mime::getheader**  
 [**3.5**](#17dp8vu)**mime::setheader**  
 [**3.6**](#3rdcrjn)**mime::getbody**  
 [**3.7**](#26in1rg)**mime::copymessage**  
 [**3.7**](#lnxbz9)**mime::buildmessage**  
 [**3.8**](#35nkun2)**smtp::sendmessage**  
 [**3.9**](#1ksv4uv)**mime::parseaddress**  
 [**3.10**](#44sinio)**mime::parsedatetime**  
 [**3.10**](#2jxsxqh)**mime::mapencoding**  
 [**3.10**](#z337ya)**mime::reversemapencoding**  
 [**4.**](#3j2qqm3)**EXAMPLES**  
 [**§**](#1y810tw)**References**  
 [**§**](#3whwml4)**Author's Address**  
 [**A.**](#2bn6wsx)**TODO List**  
 [**B.**](#qsh70q)**Acknowledgements**

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **1. SYNOPSIS**

package provide mime 1.2  
 package provide smtp 1.2

Tcl MIME is an implementation of a Tcl package that generates and parses [**MIME**](#4i7ojhp)[1] body parts.

Each MIME part consists of a header (zero or more key/value pairs), an empty line, and a structured body. A MIME part is either a "leaf" or has (zero or more) subordinates.

MIME defines four keys that may appear in the headers:

Content-Type: describes the data contained in the body ("the content"); Content-Transfer-Encoding: describes how the content is encoded for transmission in an ASCII stream; Content-Description: a textual description of the content; and, Content-ID: a globally-unique identifier for the content.

Consult [**[2]**](#2xcytpi) for a list of standard content types. Further, consult [**[3]**](#1ci93xb) for a list of several other header keys (e.g., "To", "cc", etc.)

A simple example might be:

Date: Sun, 04 July 1999 10:38:25 -0600  
 From: Marshall Rose <mrose@dbc.mtview.ca.us>  
 To: Andreas Kupries <a.kupries@westend.com>  
 cc: dnew@messagemedia.com (Darren New)  
 MIME-Version: 1.0  
 Content-Type: text/plain; charset="us-ascii"  
 Content-Description: a simple example  
 Content-ID: <4294407315.931384918.1@dbc.mtview.ca.us>  
   
 Here is the body. In this case, simply plain text.

In addition to an implementation of the mime package, Tcl MIME includes an implementation of the smtp package.

#### **1.1 Requirements**

This package requires:

* [**Tcl/Tk version 8.0.3**](http://www.scriptics.com/software/8.1.html) or later

In addition, this package requires one of the following:

* [**Trf version 2.0p5**](http://www.oche.de/~akupries/soft/trf/) or later
* [**base 64 version 2.0**](http://dev.ajubasolutions.com/software/tcllib/) or later (included with tcllib)

If it is available, Trf will be used to provide better performance; if not, Tcl-only equivalent functions, based on the base64 package, are used.

#### **1.2 Copyrights**

(c) 1999-2000 Marshall T. Rose

Hold harmless the author, and any lawful use is allowed.

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **2. SYNTAX**

[**mime::initialize**](#3dy6vkm) returns a token. Parameters:

?-canonical type/subtype  
 ?-param {key value}?...  
 ?-encoding value?  
 ?-header {key value}?... ?  
 (-file name | -string value | -parts {token1 ... tokenN})

[**mime::finalize**](#1t3h5sf) returns an empty string. Parameters:

token ?-subordinates "all" | "dynamic" | "none"?

[**mime::getproperty**](#4d34og8) returns a string or a list of strings. Parameters:

token ?property | -names?

[**mime::getheader**](#2s8eyo1) returns a list of strings. Parameters:

token ?key | -names?

[**mime::setheader**](#17dp8vu) returns a list of strings. Parameters:

token key value ?-mode "write" | "append" | "delete"?

[**mime::getbody**](#3rdcrjn) returns a string. Parameters:

?-command callback ?-blocksize octets? ?

[**mime::copymessage**](#26in1rg) returns an empty string. Parameters:

token channel

[**mime::buildmessage**](#lnxbz9) returns a string. Parameters:

token

[**smtp::sendmessage**](#35nkun2) returns a list. Parameters:

token ?-servers list? ?-ports list?  
 ?-queue boolean? ?-atleastone boolean?  
 ?-originator string? ?-recipients string?  
 ?-header {key value}?...

[**mime::parseaddress**](#1ksv4uv) returns a list of serialized arrays. Parameters:

string

[**mime::parsedatetime**](#44sinio) returns a string. Parameters:

[string | -now] property

[**mime::mapencoding**](#2jxsxqh) returns a string. Parameters:

encoding\_name

[**mime::reversemapencoding**](#z337ya) returns a string. Parameters:

mime\_charset

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **3. SEMANTICS**

#### **3.1 mime::initialize**

mime::initialize creates a MIME part:

* If the -canonical option is present, then the body is in canonical (raw) form and is found by consulting either the -file, -string, or -part option.  
    
  In addition, both the -param and -header options may occur zero or more times to specify "Content-Type" parameters (e.g., "charset") and header keyword/values (e.g., "Content-Disposition"), respectively.  
    
  Also, -encoding, if present, specifies the "Content-Transfer-Encoding" when copying the body.
* If the -canonical option is not present, then the MIME part contained in either the -file or the -string option is parsed, dynamically generating subordinates as appropriate.

#### **3.2 mime::finalize**

mime::finalize destroys a MIME part.

If the -subordinates option is present, it specifies which subordinates should also be destroyed. The default value is "dynamic".

#### **3.3 mime::getproperty**

mime::getproperty returns the properties of a MIME part.

The properties are:

property value  
 ======== =====  
 content the type/subtype describing the content  
 encoding the "Content-Transfer-Encoding"  
 params a list of "Content-Type" parameters  
 parts a list of tokens for the part's subordinates  
 size the approximate size of the content (unencoded)

The "parts" property is present only if the MIME part has subordinates.

If mime::getproperty is invoked with the name of a specific property, then the corresponding value is returned; instead, if -names is specified, a list of all properties is returned; otherwise, a serialized array of properties and values is returned.

#### **3.4 mime::getheader**

mime::getheader returns the header of a MIME part.

A header consists of zero or more key/value pairs. Each value is a list containing one or more strings.

If mime::getheader is invoked with the name of a specific key, then a list containing the corresponding value(s) is returned; instead, if -names is specified, a list of all keys is returned; otherwise, a serialized array of keys and values is returned. Note that when a key is specified (e.g., "Subject"), the list returned usually contains exactly one string; however, some keys (e.g., "Received") often occur more than once in the header, accordingly the list returned usually contains more than one string.

#### **3.5 mime::setheader**

mime::setheader writes, appends to, or deletes the value associated with a key in the header.

The value for -mode is one of:

write: the key/value is either created or overwritten (the default); append: a new value is appended for the key (creating it as necessary); or, delete: all values associated with the key are removed (the "value" parameter is ignored).

Regardless, mime::setheader returns the previous value associated with the key.

#### **3.6 mime::getbody**

mime::getbody returns the body of a leaf MIME part in canonical form.

If the -command option is present, then it is repeatedly invoked with a fragment of the body as this:

uplevel #0 $callback [list "data" $fragment]

(The -blocksize option, if present, specifies the maximum size of each fragment passed to the callback.)

When the end of the body is reached, the callback is invoked as:

uplevel #0 $callback "end"

Alternatively, if an error occurs, the callback is invoked as:

uplevel #0 $callback [list "error" reason]

Regardless, the return value of the final invocation of the callback is propagated upwards by mime::getbody.

If the -command option is absent, then the return value of mime::getbody is a string containing the MIME part's entire body.

#### **3.7 mime::copymessage**

mime::copymessage copies the MIME part to the specified channel.

mime::copymessage operates synchronously, and uses fileevent to allow asynchronous operations to proceed independently.

#### **3.7 mime::buildmessage**

mime::buildmessage returns the MIME part as a string. It is similar to mime::copymessage, only it returns the data as a return string instead of writing to a channel.

#### **3.8 smtp::sendmessage**

smtp::sendmessage sends a MIME part to an SMTP server. (Note that this procedure is in the "smtp" package, not the "mime" package.)

The options are:

-servers: a list of SMTP servers (the default is "localhost"); -ports: a list of SMTP ports (the default is 25); -queue: indicates that the SMTP server should be asked to queue the message for later processing; -atleastone: indicates that the SMTP server must find at least one recipient acceptable for the message to be sent; -originator: a string containing an 822-style address specification (if present the header isn't examined for an originator address); -recipients: a string containing one or more 822-style address specifications (if present the header isn't examined for recipient addresses); and, -header: a keyword/value pairing (may occur zero or more times).

If the -originator option is not present, the originator address is taken from "From" (or "Resent-From"); similarly, if the -recipients option is not present, recipient addresses are taken from "To", "cc", and "Bcc" (or "Resent-To", and so on). Note that the header key/values supplied by the "-header" option (not those present in the MIME part) are consulted. Regardless, header key/values are added to the outgoing message as necessary to ensure that a valid 822-style message is sent.

smtp::sendmessage returns a list indicating which recipients were unacceptable to the SMTP server. Each element of the list is another list, containing the address, an SMTP error code, and a textual diagnostic. Depending on the -atleastone option and the intended recipients,, a non-empty list may still indicate that the message was accepted by the server.

#### **3.9 mime::parseaddress**

mime::parseaddr takes a string containing one or more 822-style address specifications and returns a list of serialized arrays, one element for each address specified in the argument.

Each serialized array contains these properties:

property value  
 ======== =====  
 address local@domain  
 comment 822-style comment  
 domain the domain part (rhs)  
 error non-empty on a parse error   
 group this address begins a group  
 friendly user-friendly rendering  
 local the local part (lhs)  
 memberP this address belongs to a group  
 phrase the phrase part  
 proper 822-style address specification  
 route 822-style route specification (obsolete)

Note that one or more of these properties may be empty.

#### **3.10 mime::parsedatetime**

mime::parsedatetime takes a string containing an 822-style date-time specification and returns the specified property.

The list of properties and their ranges are:

property range  
 ======== =====  
 hour 0 .. 23  
 lmonth January, February, ..., December  
 lweekday Sunday, Monday, ... Saturday  
 mday 1 .. 31  
 min 0 .. 59  
 mon 1 .. 12  
 month Jan, Feb, ..., Dec  
 proper 822-style date-time specification  
 rclock elapsed seconds between then and now  
 sec 0 .. 59  
 wday 0 .. 6 (Sun .. Mon)  
 weekday Sun, Mon, ..., Sat  
 yday 1 .. 366  
 year 1900 ...  
 zone -720 .. 720 (minutes east of GMT)

#### **3.10 mime::mapencoding**

mime::mapencoding takes a string containing the name of a tcl encoding (see [encoding names]) and returns the MIME charset name for that encoding (or "" if the charset name is unknown).

#### **3.10 mime::reversemapencoding**

mime::reversemapencoding takes a string containing the name of a MIME charset tcl encoding (see [encoding names]) and returns the MIME charset name for that encoding (or "" if no known tcl encoding maps to the mime charset type).

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **4. EXAMPLES**

package require mime 1.0  
package require smtp 1.0  
  
  
# create an image  
  
set imageT [mime::initialize -canonical image/gif \  
 -file logo.gif]  
  
  
# parse a message  
  
set messageT [mime::initialize -file example.msg]  
  
  
# recursively traverse a message looking for primary recipients  
  
proc traverse {token} {  
 set result ""  
  
# depth-first search  
 if {![catch { mime::getproperty $token parts } parts]} {  
 foreach part $parts {  
 set result [concat $result [traverse $part]]  
 }  
 }  
  
# one value for each line occuring in the header  
 foreach value [mime::getheader $token To] {  
 foreach addr [mime::parseaddress $value] {  
 catch { unset aprops }  
 array set aprops $addr  
 lappend result $aprops(address)  
 }  
 }  
  
 return $result  
}  
  
  
# create a multipart containing both, and a timestamp  
  
set multiT [mime::initialize -canonical multipart/mixed  
 -parts [list $imageT $messageT]]  
  
  
  
  
# send it to some friends  
  
smtp::sendmessage $multiT \  
 -header [list From "Marshall Rose <mrose@dbc.mtview.ca.us>"] \  
 -header [list To "Andreas Kupries <a.kupries@westend.com>"] \  
 -header [list cc "dnew@messagemedia.com (Darren New)"] \  
 -header [list Subject "test message..."]  
  
  
# clean everything up  
  
mime::finalize $multiT -subordinates all

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **References**

|  |  |
| --- | --- |
| **[1]** | [**Freed, N.**](mailto:ned@innosoft.com) and [**N.S. Borenstein**](mailto:nsb@messagemedia.com), "[**Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies**](http://info.internet.isi.edu/in-notes/rfc/files/rfc2045.txt)", RFC 2045, November 1996. |
| **[2]** | [**Freed, N.**](mailto:ned@innosoft.com) and [**N.S. Borenstein**](mailto:nsb@messagemedia.com), "[**Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types**](http://info.internet.isi.edu/in-notes/rfc/files/rfc2046.txt)", RFC 2046, November 1995. |
| **[3]** | [**Crocker, D.**](about:blank), "[**Standard for the format of ARPA Internet Text Messages**](http://info.internet.isi.edu/in-notes/rfc/files/rfc822.txt)", RFC 822, STD 11, August 1982. |

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **Author's Address**

|  |  |
| --- | --- |
|  | Marshall T. Rose |
|  | Dover Beach Consulting, Inc. |
|  | POB 255268 |
|  | Sacramento, CA 95865-5268 |
|  | US |
| **Phone:** | +1 916 483 8878 |
| **Fax:** | +1 916 483 8848 |
| **EMail:** | [**mrose@dbc.mtview.ca.us**](mailto:mrose@dbc.mtview.ca.us) |

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **Appendix A. TODO List**

mime::initialize

* well-defined errorCode values
* catch nested errors when processing a multipart

|  |
| --- |
| [**TOC**](#gjdgxs) |

### **Appendix B. Acknowledgements**

This package is influenced by the safe-tcl package (Borenstein and Rose, circa 1993), and also by [**Darren New**](mailto:dnew@messagemedia.com)'s unpublished package of 1999.

This package makes use of [**Andreas Kupries**](mailto:a.kupries@westend.com)'s excellent Trf package.