

NAME

atmail – attachment mailer for bash (version 0.4)
 Copyright (c) 2004
 by Andreas Loesch <software@andreasloesch.de>

SYNOPSIS

atmail [*OPTIONS*] *RECIPIENT-OPTIONS* *PARAMETER*

The parameter and switches are fully described in the appropriate sections later in this manual. In detail the shell script should be called like this:

```
atmail [ -h ] [ -v ] [ -d ] [ -s subject ] [ -f sender ] [ -a header ] [ -e charset ] <recipients> message [ files... ]
```

```
atmail [ -h ] [ -v ] [ -d ] [ -s subject ] [ -f sender ] [ -a header ] [ -e charset ] <recipients> - [ files... ]
```

and the <*recipients*> are given as comma separated lists for the TO, CC and BCC recipients. Each recipient list switch can be used more than once.

DESCRIPTION

atmail is a small Bash-Script to send mails with binary attachments from the shell without other mail user agents. It is perfectly suited to be used inside administrative shell scripts. The Files are encoded as base64 so every modern MUA at the client-side should be able to decode them.

The basic idea was to build a very simple script alternative for mail or mailx with easy usage of binary attachments. So I tried to allow all of the options you could use with mail or mailx.

When you use a script for administrative tasks like sending out monthly bills you don't want to show the recipient the real user-Account you send this mail from, more often you want the client-side MUA to reply to your invoice or sales office, so you can tell **atmail** to use a different sender address in the From-line. For the same reason you can add additional mail-header fields, like "X-Bill-Number: 08/15"

There are no known limits for files you can attach (but don't use mail for excessively many or big files!)

OPTIONS

The order of the options does not matter, you can mix them as you like.

- h** show the synopsis for **atmail**. The execution is stopped after the output.
- v** Verbose mode. The details of **atmail** process are displayed on the user's terminal.
- d** Dummy mode. Execute normally but instead of piping the constructed Mail to sendmail, send it to stdout. This flag is mainly for debugging purposes.
- s** *subject*
 Set the mailsubject to *subject*. Use once, if the option is used more than one time the last occurrence is used for the mail. If you use whitespaces in the subjectline make sure to quote the parameter accordingly.
- f** *sender*
 Set the From-header to *sender*. Use once, if the option is used more than one time the last occurrence is used for the mail. If you use whitespaces in the address (e.g. My Office <office@domain.tld>) make sure to quote the parameter accordingly.
- a** *additional header*
 Add an additional header to the mail. Use one switch per additional header. If you use whitespaces in the Header make sure to quote the parameter accordingly.
- e** *encoding*
 The message body is stored as *Content-Type: text/plain; charset=<ENCODING>*. The default <ENCODING> is us-ascii, this can be overwritten with the condiguration option (see

CONFIGURATION) or with this switch. Make sure, your encoding will work, no checks are applied!

RECIPIENT-OPTIONS

The recipients of the Mail can be set by these option switches, according to the desired recipient header in the mail (TO, CC, BCC). At least one address is mandatory, but you can pass many addresses with comma separated lists or with multiple switches. Each list can be used as often as needed. If you want to use mailaddresses with Realnames (Name <name@domain.tld>) make sure to use correct quoting.

[**-t** <TO-recipientlist>]

[**-c** <CC-recipientlist>]

[**-b** <BCC-recipientlist>]

and

recipientlist := <recipient> | , <recipientlist>

recipient := <mailaddress> | "< <mailaddress> >"

| "Realname < <mailaddress> >"

mailaddress is any correct emailaddress, that sendmail can use.

PARAMETER

Together with one recipient the mail message ist mandatory. The attachments are not necessary but normally you use this script, if you want to send attachments :)

message or -

The message body can be set on the commandline use quotes to protect the whitespaces or piped into the script via stdin. With - as message text the script checks the stdin and uses the incoming text. You can also use - without piping a message to the script. In this case you will be prompted and can enter the Text manually.

files All parameters after the message text will be treated as files that should be attached to the mail. You can use absolute or relative paths. The files are checked with *file* for their mime-type and encoded as base64 with *uuencode*.

EXAMPLES

atmail -v -t man@moon.org -s "look at the pictures" "Hi man in the moon, loot at these pix" ~/sun.jpg ../fullmoon.png

here we send a Mail with the subject *look at the pictures* and the body *Hi man in the moon, loot at these pix* to man@moon.org and attach two pictures. Each step is printed to stdout, because we used the verbose mode.

atmail -v -t man@moon.org -s "look at the pictures" "Hi man in the moon, loot at these pix" ~/sun.jpg ../fullmoon.png

Here we use the **-h** switch and print the synopsis to stdout. After this the script exits and all other parameter are discarded! You see also, the order of the sequence of switches does not matter. Only the message and the files have defined positions.

atmail -h

leads to the same output as the previous examble.

```
atmail -t "Foo Bar <foo@bar.tld>,Tom <mail01@massmailer.org>rq -t paul@mary.com -c
peter@mary.com "Hi there"
```

sends a the mail to Foo Bar, Tom and paul and as cc to peter without attachments and an default subjectline

```
atmail -t "Foo Bar <foo@bar.tld>,Tom <mail01@massmailer.org>rq -f "Katie <office@mydo-
main.net>" "call me!"
```

this mail is personalized, the sender is not the Unix-useraccount but Katie from the office...

REQUIREMENTS

atmail needs some other tools to work properly.

sendmail

or a compatible mailer, important is, that the mailer can extract the recipients out of the message. If your version needs switches to work in a sendmail compatible way, you can configure it with the *SENDMAILOPS*

uuencode

to encode binary attachments.

file and also a default mime type database. **atmail** uses **file** to determine the mimetype of the attachment to set the correct Content-Type in the mail message.

CONFIGURATION

You can patch **atmail** to the local situation of your machine. In the first place you should make sure that the paths to the needed executables are correct.

TEMPPATH="/tmp"

the temporary directory

SENDMAIL="/usr/sbin/sendmail"

the path to sendmail

SENDMAILOPS=

and possible options

UUENCODE="/usr/bin/uuencode"

the path to uuencode

FILE="/usr/bin/file"

the path to file (for mimetype)

HOST=

the message-id is constructed with the current date, time, the Process-id and the user-id the domainpart is localhost or the host set here

VERBOSE='=> '

line-marker for verbose-Output

CHARSET="us-ascii"

the default charset for the message body, can be overwritten with **-e charset**

BUGS

known bugs

atmail has no know bugs, only one minor problem that should be tweaked out in one of the next releases. If you enter the message text interactively all empty lines are discarded, so if you want to construct a mail with a well formatted text you should write it first and pipe it into **atmail**.

report bugs or improvements

If you discover new bugs or have other improvement ideas, send them as mail to the author.

AUTHOR

Andreas Loesch, <mailto:software@andreasloesch.de>

COPYRIGHT and LICENSE

atmail – attachment mailer for the bash
Copyright (c) 2004, Andreas Loesch

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA