ListSERV Manual

Release 3 (8 December 2024)

Peter Simons <simons@peti.GUN.de>

1 Copyright

1.1 ListSERV is not gratis

When I started writing ListSERV a few month ago, all I needed was a simple daemon that automates the un-/subscription process. But when I released the first, simple version, I noticed that my ListSERV was the only program of its kind available for the Amiga and many, many users had requests for various features. You can guess how it continued from then on, can't you? After all, I spent several hundred hours programming and chasing down bugs with weird configurations (you won't believe how many different sendmails are out there) and finally, ListSERV has become a pretty sophisticated product. That's why I think it's fair to get at least a few bucks out of my work.

This version of ListSERV is an experimental version with only a few limitations to let you test whether you think this software satisfies your needs. This evaluation version has the FAQ and LONGINDEX commands disabled and won't allow you to administrate more than 2 lists. This should be enough for you to test the capabilities and might even serve your purposes if your needs are small.

1.2 Registration

If you think ListSERV is what you need and you want to enjoy the full capabilities, you must register with me. The shareware fee is DM 20 or US $15. If you pay an extra fee for the material (DM 10 for european users or US $10 for non-european users), you'll receive a fully printed manual and the latest version of ListSERV on floppy disk via snail mail.

Additionally, registered users will be subscribed to the support mailing list where they can get obtain assistance with setup problems or submit bug reports and enhancement requests. I will also announce or even submit new versions on this list to make sure you do not miss out a newer version.

If you want to register, please fill out the included `Registration.txt` form and deliver it to me via either e-mail or snail mail. *Please take note that I do only accept "Deutsche Mark" or "US Dollars".*

2 Concept of mailing lists

Basically, the USENET provides two kinds of communication: News and mail. While mail is of a strictly private nature and is most suitable for exchange between two or three persons, news is public and allow discussions with a greater number of participants.

Mailing lists are something inbetween. They allow for (mostly) public discussion using the e-mail interface. This may sound strange but several valid reasons exist to choose a mailing list rather than a newsgroup. The first significant advantage is that mail is a lot faster than news and thus, mailing lists have shorter turn-around times than posting to newsgroups. Additionally, mailing lists are easy to set up and later remove if no longer required, while adding a new newsgroup requires public discussion, voting, etc. . .

Furthermore, mailing lists are much more private than newsgroups. One can control who subscribes and who does not subscribe. One is even able to ban people from the list if they misbehave — what is not possible in a news environment.

So, how does this magical "mailing list" work technically? First you have to find someone willing to install the required software on his machine. Who is obviously you, since you're reading this document. This machine is called the mailing list *host*. The list stands and falls with this machine. It must be able to handle all the traffic, it should work reliable and lastly, it should have a fast connection to the net. The quicker this machine is reached by e-mail, the quicker the the turnaround for the mailing list.

Once the list is set up and announced, a list of addresses has to be collected. These people are called the *subscribers* of the mailing list. Now, whenever someone wants to *post* to the list, he simply sends an e-mail message to a special address on your site, for example "testlist@yoursite.foo.bar". When the mail arrives on your site, it is processed by the ListSERV software and a copy of the mail will be sent to *all* subscribed addresses. Basically, this is what a mailing list is all about. In case someone wants to reply to our testmail, he simply sends another e-mail to the list's address and again all subscribers will receive a copy, including the person who originally started the thread.

2.1 Features of ListSERV

In the above paragraph I used the term "processing" the mail. Now, here's a list of features that describes what I meant with "processing":

- The package contains the so called ListSERV-daemon, a tool that handles all kinds of user requests automatically. Users can subscribe or unsubscribe, they can search for certain addresses and more. . .
- ListSERV's important parts are written in ARexx and thus can be modified to fit your special requirements
- ListSERV may add a signature to any article posted to the list. Each mailing list may have its own signature, so you can state vital information there.
- ListSERV automatically sets the "Reply-To:" entry in the mail header to the list's address. Therefore, replies to articles delivered via the list will be sent back to the list again. Users can simply reply.
- ListSERV supports totally secret and encrypted mailing lists, using the Pretty Good Privacy (PGP) package. You can set up confidential mailing lists that can be used for development of commercial soft-/hardware.
- ListSERV supports FAQs.
- ListSERV allows you to archive all traffic of a mailing list.

2.2 ListSERV's system requirements

ListSERV requires OS 2.04 or later. To be specific, netsupport.library requires OS 2.04 and since ListSERV relies on netsupport.library very much... There are no further requirements — besides a working UUCP or TCP installation of course. ListSERV needs only 4k stack and very little memory.

3 Installing the ListSERV mailing list package

The installation of ListSERV is trivial, and the following may look like I over reacted since dedicated a whole chapter on the installation of ListSERV. Nevertheless:

- Get ready!
- Meditate a little bit... try to understand all the implications of installing this software package. Ask yourself whether you really need all this stuff? Does the USENET make any sense at all?
- Warm up your fingers—you'll have to type some stuff soon.
- Okay, let's go: Please type "*lha x ListSERV3_0 UUCP:*" and press the RETURN key.
- Lean back and enjoy the show...
- Feel the tension rise now!
- Feel even more tension!!
- Wow—LhA has terminated: Relax now.

That's it!

Okay, that was fun, wasn't it? You'll find a directory called `ListSERV` in your `UUCP:` directory — or wherever you decided to install ListSERV. This directory has to be assigned to the logical name "ListSERV:".

3.1 Installing the netsupport.library

The installation of the library is actually very easy. Just copy the file `netsupport.library` from `ListSERV:libs/` to your `LIBS:` directory. However, the library needs its own configfile which is the topic of the following paragraphs.

You can place the file in any path you like, you can even chose any name you want. All you have to do is to set the environment variable "NSPConfig" according to the path. The library will check, in order of precedence, for a local and a global variable and use the value as the path and name for the config file. If no "NSPConfig" variable is set, the default path is `S:NSPConfig`.

A valid config file may contain one keyword per line. Keywords *must* begin at the first column. The parameter can be separated from the keyword using either space(s) or tab(s). Lines starting with a "#" are comment lines and will be ignored.

Here's an example:

```
#
# NetSupport.Library config file
#
# $VER: NSPConfig 1.23 (22.2.94)
#

#
# where the default config file is
#
MasterConfig    UULib:Config

#
# Configuration for MakeLogEntry()
#
DefaultLog      UUSpool:Logfile, stdout
ListSERV        ListSERV:logfile, stdout
```

The above entries have the following meanings:

– MasterConfig

This is important for routines dealing with config files. The caller has the possibility to provide a NULL as filename and let the library determine the correct file. The library will use the file you specify here. This should generally be the main config file of your installation, like `UULib:Config` for UUCP or `inet:s/inet.config` for the AS225r2 package. This feature makes the programs, using the library, package-independent.

– DefaultLog

You can specify a separate logfile for each program that uses the library. However, just in case you forget it, this logfile will be used. You can specify several logfiles, separated with commas. The libraries parsing routines are still a little bit weak, so please do not try special tricks, okay? An entry of "stdout" is a magic, standing for the program's standard output stream. An entry like

```
Testname        T:testlog, stdout
```

will write the log entry to both, the file and the output window. You may also specify "CONSOLE:" for the standard error stream of the program. This is always a window and will not be affected by redirection, unlike "stdout". "PRT:" will naturally send a copy of every log entry to the printer, etc...

As mentioned above, you can specify logfile(s) for each program using the library. The program should state the keyword it uses in its manual, but usually it's just the name of the program. ListSERV, for example, can be configured like this:

```
ListSERV        ListSERV:logfile, stdout
```

4 Elements of the ListSERV package

ListSERV is not a single program, but a set of tools, scripts and programs that belong together. In the following chapter, I'd like to describe each part separately.

4.1 ListSERV daemon

The ListSERV daemon is located in `ListSERV:c/`. It is responsible for handling all commands your users may want to issue. Usually, you'll set up an alias named "listserv" that pipes the incoming mail to the daemon which expects it from standard input. Such an alias looks like this: "listserv:: `"|ListSERV:c/ListSERV"`". Just add this line to your alias file and the daemon will be installed and ready to accept commands.

ListSERV parses the mail and tries to determine the sender using the "From:" line. If no valid "From:" has been found, the mail is not processed. Any other header entry other than "From:" is ignored. The mail body may contain one of the following commands:

`HELP`
: Will send a help file `ListSERV:Help` back. You may — of course — modify this file to fulfill your requirements. A basic help file is available in ASCII and TexInfo version.

`HELP listname`
: This commands returns the description file for the specified list. This is usually a short summary of the purpose of the list.

`LIST [address]`
: Lists all mailing lists to which the given address is subscribed. If you omit the 'address' the command will assume the mailbox is in the from line.

`INDEX`
: Lists all the lists available for subscription.

`LONGINDEX`
: Lists all the lists and their descriptions.

`ADD [address] listname`
`DELETE [address] listname`
: Adds or deletes the given address to or from the list specified. Mail is sent to the address given to confirm the add or delete operation.

  Attention: Please note that it is possible to add or delete someone else's subscription to a mailing list. This facility is provided so that subscribers may alter their own subscriptions from a new or different computer account. There is therefore some potential for abuse; I have chosen to limit this by mailing a confirmation notification of any addition or deletion to the address added or deleted including a copy of the message which requested the operation. At least you can find out who's doing it to you.

  If you omit the 'address' the command will assume the mailbox that is in the From: line of the message. Note that SUBSCRIBE is a synonym for ADD; UNSUBSCRIBE for DELETE.

`DELETE-ALL [address]`
`UNSUBSCRIBE-ALL [address]`
: Unsubscribes given address from all mailing lists. Mail is sent to the address given to confirm the deletions. If you omit the 'address' the command will assume the mailbox that is in the From: line of the message.

`FAQ [listname]`
: Sends a list of "Frequently Asked Questions" for the appropriate mailing list. If the listname parameter is omitted, a list of available FAQs is sent instead.

A command must be the first word on each line in the message. Lines which do not start with a command word are interpreted as HELP. So does a message that does not contain any command. A single message may contain multiple commands; some commands may result in separate replies (namely FAQ, HELP) while others will just write something to the logfile. You will always receive a response to any command found in the mail.

Additionally, ListSERV accepts the name of a default list as command line parameter, for example: "ListSERV <mail testlist". This list will be assumed as "mailinglist" if you omit the parameter at the ADD, DELETE, HELP and FAQ commands.

This feature is usually used to set up the common "*list*-request" address on your site. Mailing to this address rather than to "listserv" reduces the chance of misuse.

One short example:

```
To: Testlist-request@foo.bar.com
Subject:

ADD myaddress@mysite
FAQ
```

will add myaddress@mysite to the "Testlist" and send testlist's FAQ back. You see, it's not possible to get the list of available FAQs when mailing to a "-request" address, because ListSERV will try to send a specific FAQ out.

4.2 ListMail

ListMail is located in `ListSERV:c/`. It will send a mail to a list of receivers. Of course almost any sendmail is able to handle this task but ListMail has a few additional features. For example, it will it recognize if the address of the poster is contained in the list and will send him a short receipt rather than the complete mail again.

ListMail expects a full RFC mail, because it is using rmail to deliver it, not sendmail! These RFC mails are generated by the ListMailFilter script described below.

ListMail tries to "batch" mail as best it can, meaning that the actual body will be transferred only once and the list of all recipients will be provided to rmail over the command line. Get it. . . ? Okay, I'll try to be more verbose. Outgoing mail is stored in batches, each consisting of three files: One C.* file and two D.* files. The C.* (command-) file file is meant for your uucico and it will tell it what to do with the two other files. (Usually just send them over to the other site.) Then you have two D.* (data-) files, one contains the actual mail and the other contains the command that should be called to handle that mail. This is something like `"rmail recipient@site.domain"` or `"rmail site.domain!recipient"`. What ListMail does is simply specifying several addresses in the second data file. Then the same mail is delivered to several recipients, without having to transfer the mail several times. Otherwise you would have one mail message per individual subscribed to the list. Imagine if you had 100 users in your mailing list and everytime someone posted a message, 100 batches of mail were created. Instead you will have one batch of mail for about 20-30 addresses. However, the amount of batched addresses is limited by the max. commandline length, which is 512 characters under OS 2.04.[1]

4.3 DoNewMailinglist

This script is located in `ListSERV:s/`. It lets you easily create a new mailing list. Just make sure that the S-bit of the file is set, call "DoNewMailinglist *listname*" and the script will initialize everything for you. Please take note that this script *must* be customized for your system! (Will be described later in the chapter "Creating a mailing list".)

[1] ListMail will respect the "MaxRMailLen" entry in UUCP's config file.

4.4 SendMailingList

This script is located in `ListSERV:s/`. Please make sure it has its S-bit set. SendMailingList will read an incoming mail from standard input and handle the whole task of processing and sending it to the recipients. The only parameter this script expects is the name of the mailing list it shall post the mail to.

Usually, this script is started by an alias, set up by the DoNewMailingList script, for example: "testlist: `"|`ListSERV:s/SendMailingList testlist`"`".

4.5 MailListFilter

This script is located in `ListSERV:s/`. Please make sure it has its S-bit set. MailListFilter does some highly sophisticated magic: It takes the incoming mail and turns it into a full-blown RFC mail, required by rmail. An envelope is created, the "Reply-To:" header field is set back to the list's address, certain filed such as "Return-Receipt-To:" are filtered, etc...

You may wonder why the address in the envelope is "*listname*-error" and not just "listname". Now that is pretty reasonable: The envelope address is used whenever the mail fails, bounces or if the mail can't be delivered. With this envelope, the mail will be sent to a human user rather than to the list, what would be fatal! (Can you spell infinite loop?)

5 Creating a new mailing list

Enough of "background knowledge", let's start with the real stuff. You can create a new mailing list using the DoNewMailinglist script. However, before you can use the script, you must customize the script to your machine. DoNewMailinglist assumes your alias file as `UULIB:mail/Aliases` and your editor as "DME". If you have another setup, please change these entries with an ordinary text editor. That's all.

Now call "DoNewMailinglist test" to create a mailing list named "test". The script will start your editor several times to let you create a few files. These files will be located in the `ListSERV:groups/<name>/` directory, where "<name>" is the name of your mailing list. In our case, the directory is called `ListSERV:groups/test/`. Here's description of each file:

`List`

This file contains the list of subscribed addresses. Please take note that it must only contain the addresses. No comments or real names are allowed. List only one address per line and avoid blank lines. You should enter only your address here and let the rest handle the daemon.

`Description`

The contents of this file will be used when an LONGINDEX or HELP <list> has been issued to the daemon. This should be a brief description of the purpose of the list.

`Introduction`

This file will be sent to all new subscribers and should contain instructions about the list, for example: "Please do not post binaries to this list!"

`FAQ`

This file can be used to maintain a list FAQ (frequently asked questions). If this file exists, it will be listed with the FAQ command of daemon and will be sent out with the "FAQ <list>" command. This file is not necessary, unlike than the previous files.

`Signature`

Here you may design a small signature that is appended to every article posted to the list. I usually state the name of list and the address of the list managers here. You should preface the actual text with a few blank lines. If this file does not exist, no signature will be appended.

`Banned`

Here you can ban certain addresses or pattern from issuing commands regarding this special list, for example ADD and DELETE. All AmigaDOS patterns are supported. You can specify one pattern per line.

I usually use the following pattern per default:

```
(~(#?@#?)|#?peti.GUN.de)
```

This will ban any address that does not contain an at ("@") and every address that ends in "peti.GUN.de". Hence, it is impossible to subscribe local addresses to the list! I know it's sad but you'll always find people who try to subscribe one mailing list to another just to annoy you. This won't happen now!

Okay, when you have edited all files, the script will terminate and the list is set up and working. If you take a look at your alias file now, you'll notice that all required aliases have been appended there. You might want to modify them for certain purposes:

`test:: "|ListSERV:s/SendMailingList test"`
: Obviously, "test" is the address you can use to post to the list. If your machine is called "peti.GUN.de", the address of the list would be "test@peti.GUN.de". SendMailingList is another script that will be discussed later.

`test-error:: listserv-manager`
: This is the address bounces will be sent to. The mailing list software will set the envelope of the outgoing mail to <listname>-error rather then <listname> to prevent bounces from beeing sent back to the list, causing an infinite loop. Usually, "listserv-manager" is an alias for postmaster.

`test-admin:: listserv-manager`
: The admin address the list's administrators can be reached under. Although most users are not smart enough to use it (They prefer sending any question they have to the world-wide list usually.) it is a nice idea IMHO. You can list several addresses here in case several people are responsible for the list.

`test-request:: "|ListSERV:c/ListSERV test"`
: This address is a special case of the "listserv" alias. Since the list's name is stated on the commandline, all commands apply to this special list. This makes it easier for the user because a simple "ADD" is enough to subscribe to the list.

5.1 Testing the ListSERV daemon

Well, everything should be set up correctly now. (Check: Have you already created the "listserv"-alias that directs mail to the daemon? If not, please do it now, we'll need it immediately.)

Start your favorite mail agent and send an e-mail to "listserv". Now put the following commands in the message body:

```
LONGINDEX
FAQ
ADD listserv test
ADD your_full_address test
DELETE simons@peti.GUN.de test
```

Let's see what should happen: The FAQ and LONGINDEX command should return the appropriate lists, the first ADD should fail because "listserv" is a local address. The second ADD should return that you're already subscribed to the list and the DELETE will fail because the stated address is not subscribed at all.

After a few seconds, you should receive a few mails in your folder. You may wonder why you receive a few of them two times: The reason is that a copy of every ADD, DELETE and LIST commands will be sent to the list's administrator. Which is you!

This is what you should get:

```
Subject: Re: your ListSERV request "LONGINDEX"

------------------------ Index of mailing lists ----------------------
test
     This list is just for internal testing purposes and I really
     doubt that you'll find it very interesting!
------------------------------------------------------------------------
```

and

```
Subject: Re: your ListSERV request "FAQ"
```

```
------------------------ Index of available FAQs ---------------------
test
----------------------------------------------------------------------
```

and

```
Subject: Re: your ListSERV request "DELETE simons@peti.GUN.de test"

Per request by your_full_address
        "DELETE simons@peti.GUN.de test"
'simons@peti.GUN.de' was NOT FOUND on the 'test' mailing list.
```

Furthermore, your logfile will contain these lines:

```
(date/time) ListSERV, your_full_address: LONGINDEX
(date/time) ListSERV, your_full_address: FAQ
(date/time) ListSERV, your_full_address: ADD listserv test
(date/time) ListSERV, listserv: address is banned--mail junked
(date/time) ListSERV, your_full_address: ADD your_full_address test
(date/time) ListSERV, your_full_address: address is banned--mail junked
(date/time) ListSERV, your_full_address: DELETE simons@peti.GUN.de test
```

5.2 Testing the list

If the above examples worked correctly for you, you're installation is complete and obviously working. If something radically different happened, you should check whether all required files exist (maybe with SnoopDOS?), whether all aliases set up correctly, etc... If that doesn't help at all, feel free to contact me via e-mail to get help!

Finally, we want to test the list itself. Start your mail agent again and send an e-mail to "test" and write whatever you like. Send the the mail off when you're ready. Okay, hard-disk-access, calculating and... finished. Now look into your header and you'll find... what is this?

Sorry, but you just learned ListSERV's receipt feature. ListSERV won't deliver a copy of the mail to the person who originally sent it to the list. Instead, it will return a short receipt with the ID of the mail, the To: address and the time it has been posted to the other receivers. The subject should be "mailing list receipt". However, your nice mail is not lost, just take a look at `ListSERV:groups/test/Log`! This file will be used as an archive of every mail posted to the list!

Now try to post another message to the mailing list, but use a different "From:" address this time. Just make one up, it doesn't matter:

```
From: smart@user.foobar.com
To: listserv
Subject: My second test post

Wow, I looove Peter Simons!
```

This time, the mail should arrive in your folder correctly.

5.3 Removing a mailing list

To remove a mailing list, just delete it's directory in `ListSERV:groups/`. Furthermore, you have to remove the list's aliases from your alias file. That is all.

5.4 Banning addresses from ListSERV completely

There are several good reason to ban certain addresses from ListSERV. Maybe someone tries to annoy you or maybe you had some bad experiences with mail loops. Because of this, ListSERV

uses a global banfile to determine whether a request should be processed or not. This file has the same format as the local banfile described earlier, but it is located in `ListSERV:Banned`.

If a from address matches one of the patterns listed in this file, the mail will be junked immediately. No command will be processed, no reply will be sent. I use this file to prevent mail loops:

```
(#?-daemon#?|#?listserv#?)
```

This entry will drop any mail comming from a mail daemon — usually a bounce — or from another ListSERV.

6 Encrypted mailing lists

Encrypted mailing lists can be required for several purposes. For discussing strictly private topics, for development of commercial software, etc... ListSERV supports encrypted mailing list's using the well-known "Pretty Good Privacy" (PGP) package. ListSERV should work with any PGP version higher than 2.2. Earlier version do not work because they were not able to encrypt a message for multiple receivers.

Attention: Although encrypted mailing lists are _very_ secure, they're no absolutely secure. ListSERV is designed to work with little or no user interaction and no system that is working automatically can be really secure. Accessing your secret ring with PGP requires you to enter a password to unlock it. Unfortunately, most people do not sit in front of their computers 24 hours a day, thus you have to save the password somewhere where PGP can access it — namely in the PGPPASS environment variable. Thus, everybody who is able to access your machine (your brother, the FBI/CIA/NSA, etc...) is able to sift through your mail traffic.

Of course this doesn't really matter in 99% of the cases, because the FBI rarely breaks into houses to find out how far your commercial GUI builder is, but if you're trying to destroy the American society or to contact the Mafia via e-mail, you should consider this.

Before you read on, you should check out the PGP manual. Better yet, use it a few weeks before you really try to set the list up. PGP is not what I call trivial and you should know the terms before you provide a sensible service as an encrypted mailing list. If you know PGP, read on.

6.1 Creating an encrypted mailing list

The basic idea of the encrypted mailing list is to create a key for your machine. Maybe calling it "Mailing lists <peti.GUN.de>" or whatever you like. Then append an additional Key-ID for every encrypted mailing list you run on your system. This will make it easier for the posters using utilities like PGPSendmail. The password you use to unlock the secret part of your key *must* be set in the environment variable PGPPASS (Please refer to the PGP documentation for further details!) to make sure ListSERV can decrypt the incoming mails. Now collect the public keys of your subscribers and send them a copy of your machine key in return. Make sure all public keys are authenticated correctly. PGP will be run in batchmode and every key that is not trusted will be rejected automatically.

Now create a directory named `secretgroups` in your `ListSERV:` directory and furthermore create a subdirectory of the name of your list, just the like in the normal directory path. The directory "secretgroups" is invisible to the ListSERV daemon. Thus, it isn't listed with the INDEX command, nobody can subscribe automatically, etc. Most people won't even know that the mailing list exists.

Secret mailing list's need no Introduction, Description and FAQ file, because ListSERV can't access them anyway. All you need is the `List` file and `Signature` is you want one. The `Log` will be created as usual.

Now that the directory is set up, add the usual aliases to your alias file:

```
<listname>:: "|ListSERV:s/SendSecureMailingList <listname>"
<listname>-error:: listserv-manager
<listname>-admin:: listserv-manager
```

Please take note that we use a different script for posting now. Also, the <listname-request> alias is not necessary anymore.

The SendSecureMailingList will use the PGPMore tool included in the PGP distribution to decrypt the incoming mail. Please make sure PGPMore is available in your command path. Then it will process the mail, copy it into the archive, append the signature and send it out via

EncListMail.[1] EncListMail is pretty much the same as the ordinary ListMail, except for that it will encrypt the mail for the recipients before it sends it out. It is necessary, that all subscribers have stated the address they're subscribed under in their key-id. Otherwise PGP won't be able to determine which key to use.

As you can see, this process has quite a number of stages: The script, pgpmore, pgp, EncListMail, EncRMail, PGP and rmail. I used pipes wherever possible to make the whole thing as fast as possible. (It would probably run *very* fast on a multi-processor system, but. . .) Nevertheless, it might be a good idea, to make the programs resident. This will speed the whole process up significantly.

6.2 Common problems

Here is a short checklist you can go through in case the encrypted mailing lists does not work as expected:

- Have you set the password to unlock the list's secret key in the PGPPASS variable or do you hand it to PGP in some other way?
- Is the password you set in PGPPASS correct? Is it really the required one for your list's key?
- Is PGPMore available in your command path?
- Are the addresses of the subscribers really listed in their key?
- Does PGP accept the keys as trusted? (You should try this by simply encrypting a short text for all subscribers manually. PGP will tell you if it doesn't accept one of the keys.)
- Do you have enough memory available?
- Do you have a PGP version 2.2 or greater? (I recommend 2.3a.3 or higher, by the way!)

[1] If you don't like the mail beeing logged unencrypted, just change the order of commands in the script.

7 The author

If you want to contact me (e-mail preferred), you may use the following addresses:

SnailMail: Peter Simons, Kaiser-Konrad-Stra 80, 53225 Bonn, Germany

Voice: +49 228 471397

E-Mail: simons@peti.GUN.de

7.1 About the author

Congratulations! Amongst the 12.42% of software users who actually bother to read the documentation, you are one of the brightest as you have apparently chosen to read the hidden gem in it: The section "About The Author".

Disclaimer[1]: Although this has not been written by Peter Simons himself, it is not necessarily more objectively than it would have been if he did it himself.

As a first approximation to the author, let us have a look at a text he wrote about himself in a list of systems in his home domain. (It may be of interest to some that his self-description has been 4.46 times as long as the actual technical data of his site.)

> I (Peter Simons) was born on Sep 4th 1973 as child of a plain supermodel and a nobel price winner and I had a very nice childhood, although it has always been some kind of nuisance to me that the people used to overlook my really notable IQ because of my extraordinarily handsome appearance.

Note for the reader: I have not known Peter as a child, but you may approximate his look of today by imagining a friendly ice bear with a full beard. (Still a very handsome ice bear, as his girl-friend would probably remark, if she bothered about "all that computer stuff" like this text.)

Although Peter is not really a computer freak - ListSERV probably owes its existence to the boring breaks between playing and watching basketball, meeting girls, going to parties, watching M*A*S*H, etc. - the adoption of his nickname "Peti" as site name for his A3000 homebox (peti.GUN.de) symbolizes the fusion of man and machine to a system of high productivity. Furthermore, the natural environment of Peter is best-suited for computer people: The stationer's shop near his home is the only one I know offering Amigas, Amiga literature and Fish disks just as natural as the more mundane things a stationer sells.

<abrupt and unreasonable break>

This "About The Author" section is shareware. If you want to know how it ends or if you have moulded an opinion about Peter Simons utilizing the information provided herein, send me all your money.

Arno Eigenwillig <arno@yaps.dinoco.de>

[1] What documentation can get along without disclaimer nowadays?

8 Acknowledgments

Amiga ListSERV is a “grown project” and many people contributed in one or the other way. Nevertheless, I’d like to single a few persons out for their help and support:

Brett Wuth
: For contributing the MailListFilter script that saved me a *lot* of time and effort.

Kai ’wusel’ Siering
: The basic structure of my mailing list package has been adopted from Kai’s private ListSERV.

Carlos Amezaga
: Thanks Carlos, you have been my most constant beta tester and I wonder how many bugs ListSERV would still have if you hadn’t used my software. Additionally, thanks a lot for proofreading my manual.

Arno Eigenwillig
: Thanks a lot for the “About the Author” section. :-)

Ralph-Thomas Aussem
: Thanks for the wonderful AmigaSMail, it works like a charm. (Especially after you corrected the bug with my pipes. :->)

9 Index

Table of Contents