GetVer

Joerg Schuchardt

COLLABORATORS

	TITLE : GetVer		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Joerg Schuchardt	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

GetVer

1.1 GetVer 4.00 Manual

GetVer 4.00

- SHAREWARE -

1.2 Exclusion of the authors liability

IMPORTANT NOTE!
THE SOFTWARE BELONGING TO THE GETVER PACKAGE IS PROVIDED "AS IS" AND
COMES WITH NO WARRANTY, EITHER EXPRESSED OR IMPLIED.
THE AUTHOR IS NOT AND WILL NOT BE LIABLE FOR ANY KIND OF DAMAGE OR
LOSS THAT MAY OCCUR DUE TO DIRECT OR INDIRECT USAGE OF THIS
SOFTWARE. YOU ARE USING THE SOFTWARE ON YOUR OWN RISK.

1.3 Shareware regulations for GetVer

SHAREWARE
From version 3.00 on, GetVer is a SHAREWARE program that, without a
keyfile, has a few restrictions:
· Before each output the text
"GetVer is SHAREWARE! Please register..."
will be shown for approximately 10 seconds. Afterwards GetVer
starts its real work.
· The buffersize is fixed to a rather small amount resulting in
reduced speed.
· Saving the ascertained versionnumber as comment , adjusting the
file date to the ascertained compilation date, as well as
defining an output media in the GUI, is not possible.
If you use GetVer after a 14 days period of testing you have to
register. After registering you'll get your personal keyfile which
will disable the described restrictions.
The registration fee is:
within Germany: 10,- DM
outside Germany: 10,- US$
There is no other currency accepted!
Simply put the appropriate amount in form of a banknote into the
filled-in and folded registration form, put the form into an
envelope, so that the money is invisible from outside and send it to
my home address.
If you want to receive your keyfile quick and safe by e-mail, then
send your public PGP key to my e-mail address.
You will receive your personal keyfile the way marked in the
registration form.
By registering you only get the right to continue using the program

itself and to use your personal keyfile. You must not sell nor hand
the keyfile over to other persons. It remains my property!
My home address: e-mail:
Joerg Schuchardt ODIN@liteline.mcnet.de
Fichtenweg 4
37181 Hardegsen
Germany

1.4 Copyright regulations for GetVer

COPYRIGHT
The GetVer 4.00 archive is only allowed to be spreaded containing
the following files which must not be manipulated:
Catalogs (dir)
English (dir)
GetVer.catalog
Português (dir)
GetVer.catalog
Deutsch (dir)
GetVer.guide GetVer.guide.info
GetVer.history GetVer.history.info
GetVer.register GetVer.register.info
Install.info
English (dir)
GetVer.guide GetVer.guide.info
GetVer.history GetVer.history.info
GetVer.register GetVer.register.info
Install.info
MWB-Icons (dir)
GetVer0 GetVer0.info
GetVer1 GetVer1.info
GetVer2 GetVer2.info
GetVer3 GetVer3.info
GetVer4 GetVer4.info
GetVer5 GetVer5.info
GetVer6 GetVer6.info
Info! Info!.info
ImageCopy_GVO
S (dir)

GetVer.paths

Deutsch.info English.info

GetVer GetVer.info

Install MWB-Icons.info

It is not allowed to sell, give, lend or otherwise let third take

possession of the GetVer keyfile.

It is not allowed to put the software into a Public Domain library,

as long as none of the following prerequisites is true:

· It is the Aminet library

(including Aminet CD-ROM, Aminet Set CD-ROM and Aminet disks)

· It is the Meeting Pearls CD-ROM library

· The price of one disk of this library doesn't exceed 5,- DM(*)

· The price of one CD-ROM of this library doesn't exceed 25,- DM(*)

(*) or the equivalent amount in another currency

Any spreading of the software together with commercial products is

only allowed with the explicit, written permission of the author.

This may also be asked at the address given at Shareware .

1.5 What is the program good for?

Description

GetVer is a replacement for the standard DOS command VERSION. It

finds out the versionnumber of a program, a library, a device, a

handler or a datatype, whether it is on disk or in memory. It can

compare file-versions and it sorts and/or forms its output on

demand. It can set the ascertained versionnumber as file-comment

and the file-date to the date of compilation (provided that it is

included in the versionstring, of course). Furthermore, GetVer

accepts the usual DOS wildcards, making it rather flexible.

Now some of you may ask:

"Why should I use GetVer? There allready exists a DOS command for

that. Why not VERSION? "

1.6 Advantages to the DOS command VERSION

Why not VERSION?

The VERSION command has some disadvantages. Namely:

· you can only check one file each run

· there are no compare functions

· it can’t be started from the workbench
All this shortcommings are removed by GetVer, and the resulting new
possibilities are installed by supplementary functions.
Nevertheless, GetVer recognizes a lot more versionnumbers because
of the more liberal way of recognizing versionstrings
(-> " How it works ").

1.7 System requirements

System requirements
Since there are used some new system functions, GetVer only works
from OS 2.0 on!
The amount of RAM needed depends on the functions GetVer shall
use, on the number of files of which the version is to be
ascertained and on the selected buffer size, of course. For smooth
running there should be at least 100 KBytes of RAM accessible.

1.8 Installation of the program

Installation
The GetVer 4.00 package contains an install script for Commodore’s
Installer. To start it, please open the directory that is named with
the desired language and doubleclick the install icon.
Users of Martin Huttenloher’s MagicWB, who want to have installed
one of the icons enclosed in the "MWB-Icons" directory, have to
doubleclick the desired icon BEFORE the install script is started!
By this the chosen program icon will appear in the "GetVer" window.
Registered users please copy their keyfile as "GetVer.key" to
ENVARC: (if the keyfile shall work immediately then also copy it to
ENV:) or into the directory containig GetVer itself.
For CLI usage please asure that the program is lying in a directory
that is embedded into the search path via the PATH command.

1.9 How it works

How it works
GetVer does not search ".info" files and by default (without FORCE
argument) it only searches executables (hex code "000003F3" as first
bytes - thus even libraries, devices, handler, fonts, ...).

It searches the program code for the string "$VER:" which by default introduces the versionstring in newer files. Additionally the code is searched for the name of the file. If no "$VER:" was found, GetVer tries to identify the versionstring by using a pattern that looks something like "#?filename#?#[0-9].#[0-9]#?". This behaviour will not neccessarily result in success and GetVer will print out a string then, that obviously is no versionstring. This happend, for example, with the program "RKick" which doesn't contain a "$VER:" string but the text "RKick [1.3|2.x]" as a description for its usage. Since it matches the pattern, GetVer interprets this text as the versionstring and prints it out. Sorry! But in order to generate a rather efficient recognition algorythm, that also recongizes versionstrings of older files, we have to pay this price. Fortunately this circumstance rather seldom comes true.

1.10 How to call it from the CLI/Shell

CLI usage

When GetVer is run from the CLI resp. the Shell, it accepts the following arguments:

Synopsis

GETVER {<Name>} [DEF|DEFAULT] [FULL] [ALL] [SORT] [FORMAT] [FORCE] [BRIEF] [CMP|COMPARE [PATH=<path>] [FILE=<file>]] [REC|RECURSIVE] [DATE] [COM|COMMENT=<mode>] [LIB|DEV|DTYPE] [BUF|BUFFER=<n>] [VER|VERSION=<n>] [REV|REVISION=<n>] [SYSTEM] [INFO]

Pattern

SOURCE/A/M,DEF=DEFAULT/S,FULL/S,ALL/S,SORT/S,FORMAT/S,FORCE/S, CMP=COMPARE/S,PATH/K,FILE/K,BRIEF/S,REC=RECURSIVE/S,DATE/S, COM=COMMENT/K,LIB/S,DEV/S,DTYPE/S,BUF=BUFFER/K/N, VER=VERSION/K/N,REV=REVISION/K/N,SYSTEM/S,INFO/S:

The meaning of the arguments is defined as follows:

Filename REC|RECURSIVE

DEF|DEFAULT DATE

FULL COM|COMMENT

ALL LIB

SORT DEV

FORMAT DTYPE

FORCE BUF|BUFFER

CMP|COMPARE VER|VERSION

PATH REV|REVISION

FILE SYSTEM

BRIEF INFO

1.11 The argument Pattern|Filename

CLI argument Pattern|Filename

The name of the file of which the versionnumber is to be ascertained. It may be given with the complete path if neccessary.

When using filenames ending in ".library", ".device", "-handler" or ".datatype", GetVer automatically searches predefined standard paths (LIBS:, DEVS:, L:, SYS:Classes/Datatypes) if the file could not be found in the given path.

You may enter one or more filenames with or without wildcards.

Example:

```
> GetVer asl.library
```

Returns the versionstring of the asl.library in LIBS:.

1.12 The argument DEF|DEFAULT

CLI argument DEF|DEFAULT

Prevents GetVer from reading the tooltypes defined in the program icon. The arguments, embedded in this icon, thus will not be used as default parameters.

Example:

```
> GetVer asl.library DEFAULT
```

1.13 The argument FULL

CLI argument FULL

The output usually ends with the versionnumber. In most cases, however, there is additional information behind it, e. g. the compilation date. Using FULL, this information will be shown, too.

Example:

```
> GetVer LIBS:asl.library FULL
```

Returns the version and the compilation date of the asl.library.

1.14 The argument ALL

CLI argument ALL

When using wildcards in filenames, GetVer usually only searches the

first matching file. Using the ALL parameter, all matching files

will be searched.

Example:

> GetVer LIBS:a#? ALL

Returns the versionstrings of all executable files in LIBS:

beginning with "a".

1.15 The argument SORT

CLI argument SORT

When checking more than one file, GetVer usually takes the files one

by one, just as they are read from the source. When using the SORT

argument, the files are checked in alphabetical order.

If there are several files to check in different paths, GetVer

first sorts for the path and within each path for the filename.

Thus all files belonging to one path are kept together.

Example:

> GetVer LIBS:[a|b|c]#? C:List ALL SORT

Lists the versions of all executable files in LIBS: beginning with

"a", "b" or "c" and of "C:List" in alphabetical order.

1.16 The argument FORMAT

CLI argument FORMAT

GetVer usually returns the ascertained versionstring. In some

cases, however, this string simply contains "Version 1.3", for

example, thus not giving information about what file was checked.

Using the FORMAT argument, GetVer forms the filename, the path in

which the file was found and the versionstring into columns.

This argument will automatically be activated when the CMP|COMPARE

argument is used.

Example:

> GetVer LIBS:xpr#? ALL FORMAT

Returns an unsorted list of the versionnumbers of all libraries

beginning with "xpr" formed into columns.

1.17 The argument FORCE

CLI argument FORCE
Without this argument GetVer only searches executables (hex code
"000003F3" as first bytes - thus even libraries, devices, handler,
fonts, ...). By using the FORCE argument, it also checks other files
(e. g. ASCII files).
Example:
> GetVer GetVer.guide FORCE
Forces GetVer to read the "GetVer.guide" which is not an executable,
of course.

1.18 The argument CMP|COMPARE

CLI argument CMP|COMPARE
In order to find out whether new libraries, devices, handler,
datatypes or other files are newer than those you allready own, you
can use the COMPARE argument. At first, it reads the file from the
given path and than from LIBS:, DEVS:, L: or SYS:Classes/DataTypes
depending on the file's type and prints out the ascertained versions
of both of them.
For easier identification of the newest file, the largest version
number is printed in colour 2 (white) while all others are printed
in colour 3 (blue).
This argument automatically activates the FORMAT argument.
Example:
> GetVer DF0:Libs/reqtools.library COMPARE
Returns the versionnumbers of the "reqtools.library" in DF0:Libs/
and in LIBS:.
Note:
· When using COMPARE together with LIB , DEV or DTYPE , the file in
the given path will be compared with the corresponding file in
memory.
· In order to change the predefined compare paths, please use the
arguments PATH or FILE .

1.19 The argument PATH

CLI argument PATH

THIS ARGUMENT IS A SUB-FUNCTION OF CMP|COMPARE AND THUS IS ONLY

AVAILABLE IN COMBINATION WITH THIS OPTION!

If you want any other path than SYS:Classes/DataTypes, LIBS:, DEVS:,

or L: to be the compare path, then you can tell GetVer to use the

desired path by entering it after the PATH keyword.

To enter more than one path is possible by separating the them by

a comma.

Example:

```
> GetVer xpkNUKE.library COMPARE PATH DF1:Libs,LIBS:compressors
```

Returns the versions of the "xpkNUKE.library" in the actual path, in

DF1:Libs and in LIBS:compressors.

1.20 The argument FILE

CLI argument FILE

THIS ARGUMENT IS A SUB-FUNCTION OF CMP|COMPARE AND THUS IS ONLY

AVAILABLE IN COMBINATION WITH THIS OPTION!

If you want any other path than SYS:Classes/DataTypes, LIBS:, DEVS:,

or L: to be the compare path, then you can use a text editor to

write those paths into a file, each line a directory.

To tell GetVer to use this file, enter it's name after the FILE

keyword.

Example:

```
> GetVer SuperTool COMPARE FILE S:GetVer.paths
```

Returns the versions of the program "SuperTool" in the actual path

as well as in all the paths standing in the file "S:GetVer.paths".

1.21 The argument BRIEF

CLI argument BRIEF

If GetVer cannot find the file in the given path, the text "'file'

not found in 'path'." appears. Since, when using the compare-

function, this output appears for every compare path in which the

file could not be found, a call may result in a rather long list in

which this very message is predominant. Therefore the argument BRIEF

offers the possibility to filter out these messages, so that outputs will only happen for existing files.

Example:

> GetVer #? COMPARE FILE S:GetVer.paths BRIEF

Returns the versions of all existing and executable files in the
actual path as well as in all the paths standing in the file
"S:GetVer.paths".

1.22 The argument REC|RECURSIVE

CLI argument REC|RECURSIVE

When using the RECURSIVE argument, GetVer not only searches the
given directory but also all its sub-directories for the filename
or pattern.

Example:

> GetVer SYS:Tools/A#? ALL RECURSIVE

Returns all ascertained versionstrings of all files beginning with
"A" in SYS:Tools and all its sub-directories.

1.23 The argument DATE

CLI argument DATE

FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!

In many files the versionstring does not only contain the version-
number but also some other information (which can be shown with the
FULL argument). If this additional information contains a date and
the DATE argument is set, GetVer sets the file-date, which you can
see by using the LIST command, to the ascertained date which usually
is the date of compilation.

If the following conditions are true, GetVer will recognize a poss-
ibly contained date string:

1. The position of day, month and year is totally unimportant. They
are read in this order and will be exchanged, if necessary:

· day against month, if the month is greater than 12 and the
day is smaller than 13.

· day against year, if the day is greater than 31 and the year
is smaller than 32.

2. The month may be embedded as number, word or short word (jan,
feb, etc.) in the german or english writing.

3. The separation of the values can be realized by blanks (" "),

points ("."), hyphens ("-") and slashes ("/").

Accepted date expressions thus would be:

17.12.94

17 Dec 94

Dez 17 94

December-17-1994

etc.

If there is a date expression embedded that doesn't meet these

rules, there will no date be recognized and the saved file-date will

not be changed.

If the attempt to save an ascertained date as the file-date fails, a

requester appears containing four buttons which (from left to right)

have the following meanings:

· Continue the search for the files' versionstrings

· Don't re-open the requester when failing again

· Switch off the DATE argument

· Cancel the operation

Example:

> GetVer xpr#?.library ALL DATE

The file-date of all XPR libraries in LIBS: containing a compilation

date is set to this ascertained date.

1.24 The argument COM|COMMENT

CLI argument COM|COMMENT

FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!

This argument safes an ascertained versionnumber as the file

comment in the form "V #.#". Afterwards you can use the LIST command

to print out the versions of all your files.

This function has three modes:

KEEP The versionnumber will only be safed as comment if

there is no other comment existing.

ADD The versionnumber will be safed as comment. A possibly

existing comment will be added.

FORCE The versionnumber will be safed as comment. A possibly

existing comment will be erased!

Each other mode name ("OFF" for example) deactivates a default

setting, possibly defined by the tooltypes.

If the attempt to save a comment fails, a requester appears con-

taining four buttons which (from left to right) have the following
meaning:
· Continue the search for the files' versionstrings
· Don't re-open the requester when failing again
· Switch off the COMMENT argument
· Cancel the operation
Example:
> GetVer LIBS: COMMENT FORCE
Lists the ascertained versions off all files in LIBS: and safes the
versionnumbers as comment of the respective file.

1.25 The argument LIB

CLI argument LIB
Using LIB, the given file will be regarded as a library and is read
from memory instead of the disk.
Example:
> GetVer exec.library LIB
Shows the version of the "exec.library" that is kept in the memory.

1.26 The argument DEV

CLI argument DEV
Using DEV, the given file will be regarded as a device and is read
from memory instead of the disk.
Example:
> GetVer timer#? DEV
Shows the version of a device standing in the memory beginning with
"timer" (usually the "timer.device").

1.27 The argument DTYPE

CLI argument DTYPE
Using DTYPE, the given file will be regarded as a datatype and is
read from memory instead of the disk.
Example:
> GetVer Amiga#? DTYPE
Shows the version of a datatype standing in the memory beginning
with "Amiga" (presumable the "Amigaguide.datatype").

1.28 The argument BUF|BUFFER

CLI argument BUF|BUFFER

FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!

Normally GetVer uses a Buffer of 2 KBytes size in which a file is loaded and then searched for the versionstring. By using BUFFER=n, where n is a number between 1 and 20 and represents the buffer size in KBytes, you can change this size.

Enlarging the buffer size will sometimes extremely reduce the execution time.

Example:

```
> GetVer LIBS:#? ALL SORT FORMAT VERSION=0 BUFFER=20
```

Sets the buffer to 20 KBytes and then lists the versions of all executable files in LIBS: that contain a versionstring in a sorted and formed way.

1.29 The arguments VER|VERSION and REV|REVISION

CLI arguments VER|VERSION and REV|REVISION

The version/revision number that the files you are about to check should at least have. If the ascertained numbers of a file are smaller, then the return code will be set to 5 (WARN). Otherwise it remains 0.

If a versionnumber is given, then the "'file': in X bytes no versionstring found" statements, shown in the case that no version string was found, will not be listed. Thus you can filter out this statements by simply adding "VER=0".

Example:

```
> GetVer LIBS:asl.library VERSION=39
```

Shows the version of the asl.library and returns WARN (5) if it is smaller than 39.

1.30 The argument SYSTEM

CLI argument SYSTEM

In order to find out the versions of the Kickstart and Workbench, use the SYSTEM argument.

Example:

```
> GetVer SYSTEM
```

Shows the version of the Kickstart and that of the Workbench.

1.31 The argument INFO

CLI argument INFO

Shows some information about GetVer.

Note!

This argument must be standing alone!

Example:

> GetVer INFO

1.32 How to call it from the Workbench

Workbench activation

There are two ways of activating GetVer from the Workbench:

1. At first select the GetVer icon(*). Then expand the selection(*) with the icons of the files of which you want to know the version number. Hold the SHIFT key pressed while doubleclicking one of the selected icons.

2. Doubleclick the GetVer icon. There will appear an AppIcon that is named "GetVer 4.00" and looks just like the GetVer icon. Now select the icons(*) of the files of which you want to know the versionnumber. Then pull the selected icons(*) onto the GetVer AppIcon and release the mouse button just when the pointer is on top of it.

To remove the AppIcon doubleclick the GetVer icon again or doubleclick the AppIcon itself and select PROJECT/QUIT from the menu of the opening GetVer preferences window.

In both cases GetVer now starts checking the files that you have selected and prints the result to the device chosen with the gadget output specification in the GUI. Usually this will be a console window which you can close by clicking the close symbol in the upper left corner or by pressing <CTRL-\>.

Workbench arguments

The Workbench variant accepts most of the arguments explained at the CLI usage, just as described there as tooltypes of the program icon. The icons included in the package contain the following tooltypes:

(FULL)

ALL

SORT

FORMAT

(FORCE)

(RECURSIVE)

(CMP)

(PATH=<path>)

(FILE=<file>)

(BRIEF)

(COMMENT=<mode>)

(DATE)

(LIB|DEV|DTYPE)

BUFFER=20

VERSION=0

OUTPUT=CON:8/24/784/320/

GetVer 4.00 Copyright © 1992-1995 by J. Schuchardt/

AUTO/WAIT/SCREEN=Workbench

Tooltypes in brackets are inactive.

For a better survey the OUTPUT argument is written in three

lines.

(*) For details about how to select a single icon, expanded selection

and pulling icons, please read your Workbench manual.

1.33 The Graphical User Interface (GUI)

The "Graphical-User-Interface" (GUI)

To temporary change the arguments predefined by the tooltypes

without manipulating the tooltypes themselves, simply doubleclick

the AppIcon. Thereupon a window opens in which you can adjust the

arguments as follows (the names written in capital letters are the

appropriate arguments of the CLI usage):

INPUT

Switches

All matching ones - ALL

Non-Executables - FORCE

Recursive - RECURSIVE

Compare - COMPARE

By marking this checkbox, two stringgadgets will be

activated. In the first one you may enter the paths to

search for the files to compare. Multiple directories

can be separated by commas without blanks. In the second

stringgadget you may enter the name of the file that
contains a list of additional compare paths. The knobs
at the ends of these gadgets, each labeled with a
question-mark, activate an ASL-filerequester, which will
help you in selecting these paths resp. this file.
Selecting the checkbox activates resp. deactivates the corresponding function.

Checking resident files

Source - LIB
DEV
DTYPE

By clicking this cycle gadget you can select whether
files from disk or from memory shall be checked.
In case you want to check files from memory you can
determine what kind of resident files to use (libraries,
devices or datatypes) by selecting the appropriate radio
button.
From the keyboard you can select a radio button with
the space bar.

OUTPUT

Switches

Full text - FULL
Sorted - SORT
Formated - FORMAT
Only on success - VERSION

This checkbox is marked if the tooltypes contain a
minimum versionnumber. Since the Workbench doesn't
interpret return codes, this function only has a
filtering effect. It is the same as setting the version
number to zero (VERSION=0). Herewith the statement
"...no versionstring found" will not be shown.

Be brief - BRIEF
Date - DATE

FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!

Selecting the checkbox activates resp. deactivates the corresponding function.

Saving the versionnumber as comment

Comment - COMMENT

FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!

With this cycle gadget you activate the COMMENT function
by selecting one of the three possible modes KEEP, ADD
and FORCE.
Selecting the output media
Output specification
FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!
If you neither like the window predefined by the
tooltype OUTPUT, nor the standard window of the program,
you can adjust the ouput media with this string gadget.
The output may be redirected to any available device
(CON:, PRT:, DHx:, DFx:, NIL:, etc.). When redirecting
into a non-existing file, it will be generated.
Afterwards or if the file was allready existing, new
results will be added at the end of the file.
Adjusting the buffer size
Buffer size (KB) - BUFFER
FUNCTION NOT AVAILABLE IN THE UNREGISTERED VERSION!
The buffer size can either be entered into the small string
gadget or be selected by draging the black box inside the
slider gadget. Therefore, click the small box inside the
long, horizontal frame and move the mouse to the left or
right while holding the button pressed. The box will follow
your movements within the frame where its position
represents the size of the buffer in KByte: full left means
1 KB, full right means 20 KB. The actual buffer size will be
shown in the small string gadget.
All functions, except selecting the type of files which shall be
checked in the memory (-> "Input/Checking resident files"), can be
used by pressing the underlined key.
The menu of the preferences window
Information...
Shows a short information about the program and its legal
state.
Quit
Removes the AppIcon and ends the program completely.
Each Amiga user, however, who has at least little experience, should
be able to understand the meaning of the gadgets and should know how
to use them even without explanation.

1.34 Where to get Updates

How to get an update

From version 4.00 on, GetVer will be published in the Aminet.

Users who have access to the Aminet via communication networks can

obtain the latest version from this net.

But even other users can profit by this way of publishing: through

the Aminet CD-ROM or the new Aminet disk series.

1.35 The future of GetVer

Plans for the future

I'm making up my mind about making the new function for defining

variable compare paths in a special text file more intelligent. In

detail the text file could contain the typical ending of a filetype

in the first line and the path to compare with when the file to

check has that special ending in the following line.

Example:

```
.library
LIBS:,LIBS:compressors
.device
DEVS:
.
.
.
etc.
```

What do you say? Is this idea worth a further growing of GetVer?

1.36 Whom I have to thank

Thanks

I would like to thank some persons, which more or less participated

in the development of GetVer or supported it's realization:

Volkmar Mai, who had good ideas for improving the program

right after its first publication.

Maik Wieland, who did a little beta-testing and placed his

e-mail address at my disposal at the time I

started the project and thus enabled users of

former versions to contact me.

Jürgen Urbanek, who’s ideas are responsible for most changes

since version 3.00.

Nils Heeren, who is responsible for the translation of the

catalog file into the português language and at

the same time did a little beta-testing.

Maxon Computer GmbH, for their MaxonC++ 3 Compiler.

And all registered users, of course.

1.37 Index

INDEX