

# **The MGR Window System HOWTO**

# Table of Contents

<b><u>The MGR Window System HOWTO</u></b> .....	<b>1</b>
<u>Vincent Broman</u> .....	1
<u>1. This HOWTO</u> .....	1
<u>2. What is the MGR window system?</u> .....	1
<u>3. Installing MGR</u> .....	1
<u>4. Running MGR</u> .....	1
<u>5. Programming for MGR</u> .....	1
<u>6. More documentation</u> .....	1
<u>7. Credit for MGR</u> .....	1
<u>1. This HOWTO</u> .....	1
<u>1.1 Archiving</u> .....	2
<u>1.2 Authentication</u> .....	2
<u>1.3 Credit for the HOWTO</u> .....	2
<u>2. What is the MGR window system?</u> .....	2
<u>2.1 Function</u> .....	2
<u>2.2 Requirements</u> .....	2
<u>2.3 How do MGR, X11, and 8.5 compare?</u> .....	3
<u>3. Installing MGR</u> .....	3
<u>4. Running MGR</u> .....	6
<u>4.1 Applications not aware of MGR</u> .....	8
<u>4.2 MGR Applications (clients) distributed with the server</u> .....	8
<u>4.3 MGR-aware clients distributed separately, see "SUPPORT" file</u> .....	13
<u>5. Programming for MGR</u> .....	14
<u>6. More documentation</u> .....	15
<u>7. Credit for MGR</u> .....	15

# The MGR Window System HOWTO

Vincent Broman

v0.1, 1996-05-30

---

*Information on the installation, configuration and running of the MGR Window System.*

---

**Archived Document Notice:** This document has been archived by the LDP because it does not apply to modern Linux systems. It is no longer being actively maintained.

## 1. [This HOWTO](#)

- [1.1 Archiving](#)
- [1.2 Authentication](#)
- [1.3 Credit for the HOWTO](#)

## 2. [What is the MGR window system?](#)

- [2.1 Function](#)
- [2.2 Requirements](#)
- [2.3 How do MGR, X11, and 8.5 compare?](#)

## 3. [Installing MGR](#)

## 4. [Running MGR](#)

- [4.1 Applications not aware of MGR](#)
- [4.2 MGR Applications \(clients\) distributed with the server](#)
- [4.3 MGR-aware clients distributed separately, see "SUPPORT" file](#)

## 5. [Programming for MGR](#)

## 6. [More documentation](#)

## 7. [Credit for MGR](#)

---

## 1. [This HOWTO](#)

Copyright Vincent Broman 1995.  
Permission granted to make and distribute copies of this HOWTO  
under the conditions of the GNU General Public License.

## 1.1 Archiving

This HOWTO is archived in `ftp://archimedes.nosc.mil/pub/Mgr/MGR-HOWTO.sgml`, and also distributed from `ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/MGR-HOWTO`. In nearby directories the same document may appear in alternate formats like `MGR-HOWTO.txt`.

## 1.2 Authentication

Copies of the MGR distribution due to Broman should be accompanied by PGP signature files, signed by "Vincent Broman <broman@nosc.mil>".

## 1.3 Credit for the HOWTO

While Vincent Broman first put together this HOWTO, much of the information and text was obtained from FAQs, READMEs, etc. written by Stephen Uhler, Michael Haardt, and other public-spirited net-persons. Email corrections and suggested changes to `broman@nosc.mil`.

Uhler was the main architect of **MGR** — see the Credit section below.

---

## 2. [What is the MGR window system?](#)

### 2.1 Function

**MGR** (ManaGeR) is a graphical window system. The **MGR** server provides a builtin window manager and windowed graphics terminal emulation on color and monochrome bitmap displays. **MGR** is controlled by mousing pop-up menus, by keyboard interaction, and by escape sequences written on pseudo-terminals by client software.

**MGR** provides each client window with: termcap-style terminal control functions, graphics primitives such as line and circle drawing; facilities for manipulating bitmaps, fonts, icons, and pop-up menus; commands to reshape and position windows; and a message passing facility enabling client programs to rendezvous and exchange messages. Client programs may ask to be informed when a change in the window system occurs, such as a reshaped window, a pushed mouse button, or a message sent from another client program. These changes are called events. **MGR** notifies a client program of an event by sending it an ASCII character string in a format specified by the client program. Existing applications can be integrated into the windowing environment without modification by having **MGR** imitate keystrokes in response to user defined menu selections or other events.

### 2.2 Requirements

**MGR** currently runs on Linux, FreeBSD, Sun 3/4 workstations with SunOS, and Coherent. Various older versions of **MGR** run on the Macintosh, Atari ST MiNT, Xenix, 386-Minix, DEC 3100, and the 3b1 Unix-pc. Many small, industrial, real-time systems under OS9 or Lynx in Europe use (another variant of) Mgr for their user interface. The programming interface is implemented in C and in ELisp, although supporting clients written in other languages is quite easy.

Running **MGR** requires much less in resources than X, or even gcc. It does not have the user-base, software repertory, or high-level libraries of X or MS-Windows, say, but it is quite elegant and approachable.

It has been said that **MGR** is to X as Unix was to Multics.

## 2.3 How do MGR, X11, and 8.5 compare?

**MGR** consists of a server with builtin window manager and terminal emulator, and clients which run in this terminal emulator and use it to communicate with the server. No resource multiplexing is done.

X11 consists of a server and clients, which usually connect to the server using a socket. All user visible things like terminal emulators, window managers etc are done using clients. No resource multiplexing is done.

8.5, the Plan 9 window system, is a resource multiplexer, as each process running in a window can access /dev/bitblt, /dev/mouse and /dev/kbd in its own namespace. These are multiplexed to the /dev/bitblt, /dev/mouse and /dev/kbd in the namespace of 8.5. This approach allows one to run 8.5 in an 8.5 window, a very clean design. 8.5 further has an integrated window manager and terminal emulator.

---

## 3. Installing MGR

The latest source distribution can be FTPed from the directory `ftp://archimedes.nosc.mil/pub/Mgr/69` or Mosaiced from `http://archimedes.nosc.mil/Mgr/69`. The same should be found at `ftp://sunsite.unc.edu/pub/Linux/apps/MGR` and its mirrors. Older versions of this distribution from Haardt can be found on `tsx-11.mit.edu` and perhaps elsewhere. Pre-Linux versions of **MGR** from Uhler and others have been found at `ftp://bellcore.com/pub/mgr`, but I think they are gone now. I have saved a copy of everything about **MGR** seen on the Internet, but I am not aware of anything weighty that is missing from this Linux/Sun distribution. **MGR** has been through a lot of versions and releases, but the current \*Linux\* version number is 0.69. This version number could jump to 1.0 when stable 256-color VGA code for Linux appears (for more than one video card type). RCS version numbers have increased from Bellcore's 4.3 up to our 4.13 now.

Required tools to build this distribution of **MGR** are m4 (GNU, or perhaps another supporting the -D option), make (GNU, or perhaps another supporting include) and \*roff for the docs. Also sh, awk, and POSIX install. Binary distributions are not assembled often so you need an ANSI C compiler environment, e.g. gcc.

A Linux installation requires Linux 0.99.10 or better (1.2.13 is what I actually test on now), an HGC, EGA, VGA, or SVGA graphics card, and a mouse. Mouses supported are: serial Microsoft mouse, serial MouseSystems 3 and 5 byte mouse, serial MMSeries mouse, serial Logitech mouse, PS/2 mouse, or a bus mouse. With Buckey (Meta) hot keys enabled, even a mouseless system could do a certain amount of useful work under **MGR**. The VGA 640x480 monochrome graphics mode is supported out of the box, as is 640x350 and 640x200. To run 800x600, or other modes that your BIOS can initialize and which do not require bank-switching, you need to run a small program (supplied as `src/vgamisc/regs.exe`) under DOS or an emulator to read the VGA registers and write a header file which you place in the directory `src/libbitblit/linux`, so that it can be #include'd by the `vga.c` file there. Samples of these files are supplied, but please create your own. Some VGA cards can use 128k windows, and these might run higher monochrome resolutions.

## The MGR Window System HOWTO

The Linux-colorport code also runs in the standard 320x200x256 color VGA mode without difficulty, because no bank switching is required. If you think of how few 64000 pixels is, you would realize this color mode is quite limited. Non-fast, but simple, bank-switching code has been added in version 0.65, and it works with a Tseng ET4000 card in 640x480x256 and 800x600x256 modes. The S3 code does not work in super VGA resolutions, yet. Supporting new super VGA cards requires writing one function to switch banks and then making sure that the desired screen mode can be initialized from a register dump, possibly with hand-tweaking. The Linux color servers generally mangle the screen fonts, necessitating use of `restorefont` as in `runx`. If someone were to extract the VGA initialization code out of X, this might make MGR work on a lot more color systems.

Suns with SunOS 4.1.2+ and `bwtwo`, `cgthree`, or `cgsix` frame buffers are supported. Their speed handling color is good. Coherent installations should refer to the `VERSIONS/README.Coh` file in the source distribution. Porting the latest-and-greatest **MGR** to another POSIX-like system which provides `select()` and `pty`'s and direct access to a bitmapped frame-buffer ought to be straightforward, just implementing the `libbitblit` library based on the `sunmono` or `colorport` code, say.

If you want to install everything, you need 7 MB disk space for binaries, fonts, manual pages etc. The sources are about 4.5 MB, plus object files during compilation.

Normally, `/usr/mgr` should be either the directory or a link to the directory where you install **MGR** stuff for runtime use. Typing

```
cd /usr/mgr; tar xvfz whereveryouputit/mgrusr-0.69.tgz
```

and optionally

```
cd /usr/mgr; tar xvfz wherever/morefonts-0.69.tgz
```

will unpack these. The source can be put anywhere, e.g. typing

```
cd /usr/src/local/mgr; tar xvfz wherever/mgrsrc-0.69.tgz
```

to unpack the sources from `archimedes.nosc.mil`.

The source tree can be compiled from one top-level Makefile which invokes lower-level Makefiles, all of which "include" a "Configfile" at the top level. The Configfile is created by an interactive sh script named `Configure`, which asks you questions, then runs `m4` on a `Configfile.m4`. So you type something like this:

```
chdir /usr/src/local/mgr
sh ./Configure
make first
make depend
make install
make clean
```

It might be wise, before running `make`, to eyeball the Configfile generated by the `Configure` script, checking that it looks reasonable. (At least one `m4` poops out (Sun `/usr/bin/m4`), creating a very short Configfile. If this happens, try hand editing a copy of `Configfile.sun` or `Configfile.lx`) One can also make `all` in any directory with a Makefile as soon as the libraries have been compiled and installed. The server, libraries, and some clients have been linted, but several clients are K&R C code that generates many compiler warnings.

## The MGR Window System HOWTO

Several flags in MGRFLAGS can be added/omitted in the Configfile to change some optional features in the server, viz:

### ***-DWHO***

muck utmp file so "who" works

### ***-DVI***

code for clicking the mouse in vi moving the cursor

### ***-DDEBUG***

enable debugging output selectable with -d options.

### ***-DFASTMOUSE***

XOR the mouse track

### ***-DBUCKEY***

for hot-key server commands without mousing

### ***-DPRIORITY***

for priority window scheduling instead of round-robin; the active window gets higher priority

### ***-DCUT***

for cut/paste between windows and a global snarf buffer

### ***-DMGR\_ALIGN***

forces window alignment for fast scrolling (monochrome)

### ***-DKILL***

kills windows upon tty i/o errors

### ***-DSHRINK***

use only some of the screen (\$MGRSIZE in environment)

### ***-DNOSTACK***

don't permit event stacking

### ***-DBELL***

audibly ring the bell

# The MGR Window System HOWTO

## **-DKBD**

read `mgr` input from the sun kbd, instead of `stdin`. This permits redirection of console msgs to a window.

## **-DFRACCHAR**

fractional character movement for proportional fonts

## **-DXMENU**

extended menu stuff (experimental)

## **-DMOVIE**

movie making extension which logs all operations to a file for later replay — not quite working under Linux

## **-DEMUMIDMSBUT**

Emulate a missing middle mouse button by chording

Not all combinations of these options have been tested on all systems.

The `BITBLITFLAGS` macro should contain `-DBANKED` if you're trying out the super VGA color.

C code for the static variables in the server containing icons and fonts is generated by a translator from icon and font files.

Not all the clients are compiled and installed by the Makefiles. Clients found under `src/clients` having capitalized names or not compiled by the supplied Makefiles may have problems compiling and/or running, but they may be interesting to hack on. Most of the screen drivers found under the `libbitblit` directory are of mainly archeological interest. Grave robbing can be profitable.

At some point check that your `/etc/termcap` and/or `terminfo` file contain entries for **MGR** terminals such as found in the `misc` directory. If all your software checks `$TERMCAP` in the environment, this is not needed, as long as you run `eval `set_termcap`` in each window.

**MGR** works better if run `setuid root`, because it wants to `chown` ptys and write in the `utmp` file. This helps the ify iconifier client work better and the event passing mechanism be more secure. On Linux, root permissions are *required* in order to do in/out on the screen device. Otherwise, you decide whether to trust it.

In versions around 0.62 there are troubles on the Sun with using the `cs` as the default shell. Programs seem to run in a different process group than the foreground process group of the window's pty, in contradiction to man pages and posix specs. There is no trouble with `bash`, `sh`, or `rc`. Ideas why?

---

## **4. Running MGR**

## The MGR Window System HOWTO

The only file *required* in an **MGR** installation is the server itself. That would give you terminal emulator windows with shells running in them and cutting and pasting with the mouse, but no nice clocks, extra fonts, fancy graphics, etc. Depending on options, a monochrome server needs about 200K of RAM plus dynamic space for windows, bitmaps, etc.

If `/usr/mgr/bin` is in your `PATH`, then just type "mgr" to start up. After enjoying the animated startup screen, press any key. When the hatched background and mouse pointer appear, hold down the left mouse button, highlight the "new window" menu item, and release the button. Then drag the mouse from corner to corner where you want a window to appear. The window will have your default shell running in it. Hold down the left mouse button over an existing window to see another menu for doing things to that window. Left-clicking on an obscured window raises it to the top. The menu you saw that pops-up over the empty background includes the quit command. For people with a two button mouse: press both buttons together to emulate the missing middle button used by some clients.

The quit submenu includes the "really quit" option, a suspend option which should only be used if you run a job-control shell, and a screen saver and locker option, which waits for you to type your login password when you come back to your machine.

When trying to run **MGR**, if you get:

### *can't find the screen*

make sure you have a `/dev` entry for your display device, e.g. on a Sun `/dev/bwtwo0`. If not, as root cd to `/dev`, and type "MAKEDEV bwtwo0". Otherwise, you might need the `-S/dev/bwtwo0` or (on Linux) the `-S640x480` command line option when starting `mgr`. On Linux, you might also make sure that `/usr/mgr/bin/mgr` was installed setuid root.

### *can't find the mouse*

make sure `/dev/mouse` exists, usually as a symbolic link to the real device name for your mouse. If you haven't permission to write in `/dev`, then something like a `-m/dev/cua0` option can be given when starting `mgr`. Also, make sure you've supplied the right mouse protocol choice when you configured `mgr`. The mouse may speak Microsoft, even if that is not the brand name.

### *can't get a pty*

make sure all of `/dev/[tp]ty[pq]?` are owned by root, mode 666, and all programs referenced with the "shell" option in your `.mgrc` startup file (if any) exist and are executable.

### *none but the default font*

make sure **MGR** is looking in the right place for its fonts. Check the `Configfile` in the source or see whether a `-f/usr/mgr/font` option to `mgr` fixes the problem.

### *completely hung (not even the mouse track moves)*

login to your machine from another terminal (or `rlogin`) and kill the `mgr` process. A buckey-Q key can quit **MGR** if the keyboard still works.

## 4.1 Applications not aware of MGR

Any tty-oriented application can be run in an **MGR** window without further ado. Screen-oriented applications using `termcap` or `curses` can get the correct number of lines and columns in the window by your using `shape(1)` to reshape the window or using `set_termcap(1)` to obtain the correct `termcap` entry.

## 4.2 MGR Applications (clients) distributed with the server

### ***bdfmogr***

converts some BDF fonts to MGR fonts

### ***browse***

an icon browser

### ***bury***

bury this window

### ***c\_menu***

vi menus from C compiler errors

### ***clock***

digital display of time of day

### ***clock2***

analog display of time of day

### ***close***

close this window, iconify

### ***color***

set the foreground and background color for text in this window

### ***colormap***

read or write in the color lookup table

### ***cursor***

change appearance of the character cursor

### ***cut***

## The MGR Window System HOWTO

cut text from this window into the cut buffer

### *cycle*

display a sequence of icons

### *dmgr*

crude ditroff previewer

### *fade*

fade a home movie script from one scene to another

### *font*

change to a new font in this window

### *gropbm*

a groff to PBM driver using Hershey fonts

### *hpmgr*

hp 2621 terminal emulator

### *ico*

animate an icosahedron or other polyhedron

### *iconmail*

notification of mail arrival

### *iconmsgs*

message arrival notification

### *ify*

iconify and deiconify windows

### *loadfont*

load a font from the file system

### *maze*

a maze game

### *mclock*

## 4.1 Applications not aware of MGR

micky mouse clock

***menu***

create or select a pop-up menu

***mgr***

bellcore window system server and window manager

***mgrbd***

boulder-dash game

***mgrbiff***

watch mailbox for mail and notify

***mgrload***

graph of system load average

***mgrlock***

lock the console

***mgrlogin***

graphical login controller

***mgrmag***

magnify a part of the screen, optionally dump to file

***mgrmail***

notification of mail arrival

***mgrmode***

set or clear window modes

***mgrmsgs***

message arrival notification

***mgrplot***

Unix "plot" graphics filter

***mgrsclock***

sandclock

***mgrshowfont***

browse through mgr fonts

***mgrsketch***

a sketching/drawing program

***mgrview***

view mgr bitmap images

***mless***

start up less/more in separate window, menu added for less

***mnew***

startup up any program in a separate, independent window

***mphoon***

display the current phase of the moon

***mvi***

start up vi in a separate window, with mouse pointing

***oclose***

(old) close a window

***omgrmail***

(old) notification of mail arrival

***pbmrawtomgr, pgmrawtomgr, ppmrawtomgr***

convert raw PBM/PGM/PPM image files to mgr bitmap format

***pbmstream***

split out a stream of bitmaps

***pbmtpprt***

printer output from PBM

***pgs***

ghostscript patch and front end, a PS viewer

## ***pilot***

a bitmap browser, or image viewer

## ***resetwin***

cleanup window state after client crashes messily

## ***rotate***

rotate a bitmap 90 degrees.

## ***screendump***

write graphics screen dump to a bitmap file

## ***set\_console***

redirect console messages to this window

## ***set\_termcap***

output an appropriate TERM and TERMCAP setting

## ***setname***

name a window, for messages and iconifying

## ***shape***

reshape this window

## ***square***

square this window

## ***squeeze***

compress mgr bitmap using run-length encoding

## ***startup***

produce a skeleton startup file for current window layout

## ***texmgr***

TeX dvi file previewer

## ***text2font, font2text***

convert between mgr font format and text dump

***unsqueeze***

uncompress mgr bitmap using run length encoding

***vgafont2mgr, mgrfont2vga***

convert between mgr font format and VGA

***window\_print***

print an image of a window

***zoom***

an icon editor

***bounce, grav, grid, hilbert, mgreyes, stringart, walk***

graphics demos

## **4.3 MGR-aware clients distributed separately, see "SUPPORT" file**

***calctool***

on-screen calculator

***chess***

frontend to `/usr/games/chess`

***gnu emacs***

editor with `lisp/term/mgr.el` mouse & menu support

***gnuplot***

universal scientific data plotting

***metafont***

font design and creation

***origami***

folding editor

***pbmplus***

4.3 MGR-aware clients distributed separately, see "SUPPORT" file

portable bitmap format conversions, manipulations

## *plplot*

slick scientific data plotting

The Emacs support in `misc/mgr.el` and `misc/mailcap` includes very usable MIME support, via Rmail and metamail.

A general image viewer could be cobbled together from `pilot` and the `netPBM` filters, but I have not taken the time to do it.

---

## 5. Programming for MGR

The **MGR** programmers manual, the C language applications interface, is found in the `doc` directory in `troff/nroff` form. It covers general concepts, the function/macro calls controlling the server, a sample application, with an index and glossary.

Porting client code used with older versions of **MGR** sometimes requires the substitution of

```
#include <mgr/mgr.h>
```

for

```
#include <term.h>
#include <dump.h>
```

and clients using old-style `B_XOR`, `B_CLEAR`, et al instead of `BIT_XOR`, `BIT_CLR`, et al can be accommodated by writing

```
#define OLDMGRBITOPS
#include <mgr/mgr.h>
```

Compiling client code generally requires compiler options like the following.

```
-I/usr/mgr/include -L/usr/mgr/lib -lmgr
```

One can get some interactive feel for the **MGR** server functions by reading and experimenting with the `mgr.el` terminal driver for GNU Emacs which implements the **MGR** interface library in ELisp.

The usual method of inquiring state from the server has the potential of stumbling on a race condition if the client also expects a large volume of event notifications. The problem arises if an (asynchronous) event notification arrives when a (synchronous) inquiry response was expected. If this arises in practice (unusual) then the **MGR** state inquiry functions would have to be integrated with your event handling loop.

The only major drawing function missing from the **MGR** protocol, it seems, is an area fill for areas other than upright rectangles. There is new code for manipulating the global colormap, as well as (advisory) allocation and freeing of color indices owned by windows.

If you are thinking of hacking on the server, you can find the mouse driver in `mouse.*` and `mouse_get.*`, the grotty parts of the keyboard interface in `kbd.c`, and the interface to the display in the `src/libbitblit/*` directories. The main procedure, much initialization, and the top level input loop are in `mgr.c`, and the interpretation of escape sequences is in `put_window.c`.

---

## 6. [More documentation](#)

The programmer's manual is essential for concepts.

Nearly all the clients supplied come with a man page which is installed into `/usr/mgr/man/man1` or `man6`. Other useful man pages are `bitblit.3`, `font.5`, and `bitmap.5`. There is some ambiguity in the docs in distinguishing the internal bitmap format found in your frame-buffer and the external bitmap format found in files, e.g. icons.

The `mgr.1` man page covers command line options, commands in the `~/ .mgrc` startup file, mouse and menu interaction with the server, and hot-key shortcuts available on systems with such hot-keys.

Many of the fonts in `/usr/mgr/font/*` are described to some extent in `/usr/mgr/font/*.txt`, e.g. `/usr/mgr/font/FONTPATH.txt` gives X-style font descriptions for the fonts obtained in `.bdf` format. Font names end in `WxH`, where `W` and `H` are the decimal width and height in pixels of each character box.

---

## 7. [Credit for MGR](#)

Stephen Uhler, with others working at Bellcore, was the original designer and implementer of **MGR**, so Bellcore has copyrighted much of the code and documentation for **MGR** under the following conditions.

```
* Permission is granted to copy or use this program, EXCEPT that it
* may not be sold for profit, the copyright notice must be reproduced
* on copies, and credit should be given to Bellcore where it is due.
```

One required showing of the copyright notice is the startup title screen.

Other credits to:

- Stephen Hawley for his wonderful icons.
- Tommy Frandsen for the VGA linux library.
- Tom Heller for his Gasblit library.
- Andrew Haylett for the Mouse driver code.
- Dan McCrackin for his `gasblit->linux` patches.
- Dave Gymer, `dgymer@gdcarc.co.uk`, for the Startrek effect fix.
- Alex Liu for first releasing a working Linux version of **MGR**.
- Lars Aronsson (`aronsson@lysator.liu.se`) for `text2font` and an ISO8859-1 8-bit font.
- Harry Pulley (`hcpiv@grumpy.cis.uoguelph.ca`, `hcpiv@snowwhite.cis.uoguelph.ca`) for the Coherent port.
- Vance Petree & Grant Edwards & Udo Munk for their work on Hercules.
- Udo Munk for his work on serial mouse initialization & select.
- Norman Bartek & Hal Snyder at Mark Williams Co. for their help with some bugs & with Coherent device drivers.

## The MGR Window System HOWTO

- Extra thanks to Zeyd Ben Halim for lots of helpful patches, especially the adaptation of selection.
- Bradley Bosch, brad@lachman.com, for lots of patches from his 3b1 port, which fix bugs and implement new and desirable features.
- Andrew Morton, applix@runxtsa.runx.oz.au, who first wrote the cut-word code.
- Kapil Paranjape, kapil@motive.math.tifr.res.in, for the EGA support.
- Michael Haardt for MOVIE support fixes, bug fixes, separation of the libbitblit code into output drivers, expansion of the libmgr, and origami folding of the code.
- Yossi Gil for many fonts.
- Carsten Emde, carsten@thlmak.pr.net.ch, for mphoon.
- Vincent Broman for middle mouse-button emulation, linting, Sun cgsix support, VGA colormap access, integration of the sunport code into Haardt's layering scheme, font gathering, the screen saver, and continued maintenance.
- Kenneth Almquist, ka@socrates.hr.att.com, for helpful bug reports.
- Tim Pierce, twpierce@midway.uchicago.edu, for the port to FreeBSD 2.0R with Trident VGA.

All bitmap fonts from any source are strictly public domain in the USA. The 583 fixed-width fonts supplied with **MGR** were obtained from Uhler, the X distribution, Yossi Gil, and elsewhere. The Hershey vector fonts and the code for rendering them are probably freely redistributable.

---