Madhouse

Carsten Jahn

COLLABORATORS			
	TITLE : Madhouse		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Carsten Jahn	July 20, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Madhouse

1.1 Madhouse: Main page

```
  Have you got a screen blanker? Yes? Here is another one.
Out of hundreds of screen blankers, there are three or four which are
modulary with different blanker-modules. They all have some nice
effects, but they all have some very stupid one's.
Here is a modulary screen blanker which has many good effects, and
other effects will follow. The main-program, Madhouse, is *VERY* con-
figureable and has a lot of options. But look at them yourself. To see
all the effects and to start working with Madhouse, read the little
Workshop.
                                Now enjoy!
                                ~~~~~~~~~~
                      The Workshop: the quick start.
               The Advanced Options: disks & passwords.
                     The Blankmodes: all the options...
                  The Error System: fun with errors.
                The reference-part: How worked this gadget?
    !!              For disk-users: All about Buffering           !!
        Not only for C-programmers: Adding own modules
                    Troubleshooting: Known problems
                The authors and the: Copyright+Registration
```

1.2 The workshop

```
All right, you have pressed the right button to have fun with Madhouse.
Now, we explore the functions of Madhouse together.
0. Installation.
   The installation can be done completely by the Installerscript, which
   is part of this distribution. All you have to do is double-click on
   the icon named "Install". But this will only work if the program
   "Installer", which executes the script, is in your C: directory. If
   this is not the case, you can search it on your "Install"-Disk (if you
   have one) or on the disks of other programs which submit an Installer-
   script, and copy it to C:.
   If you have no possibility to get the program (I don't expect so),
   then you have to install Madhouse "by hand".
```

a) For a quick test of Madhouse: just copy Madhouse/libs/amos.library to libs:. (Perhaps you'll have to complete the path in front of "Madhouse".) If you now run Madhouse from Workbench out of its original directory, it will find the MadhouseConfigEd-program itself.
b) For permanent usage: as the word "permanent" sais, the WBStartup-Drawer will be used. So copy the program "Madhouse" to SYS:WBStartup, and copy the amos.library to libs: (see section a). All programs in WBStartup are started everytime you boot your computer.
If you have a harddisk, make a new drawer on it. Users without one will have to format a new disk... Now copy the desired files (the doc of your language, the icon you prefer) and the MadhouseConfigEd to the new drawer or the disk. The background-process "Madhouse" will start the MadhouseConfigEd itself, if this is necessary or if you want so. Don't start it yourself, but make a start by Madhouse possible. So select the icon of the Madhouse in your WBStartup-drawer, and call the menuitem "Icon/Information". Change the Tooltype "CONFIGED=": insert the complete path and the name of MadhouseConfigEd after the "=". Do not use spaces. If your ConfigEd is in Work:Madhouse, the line has to look like this:
CONFIGED=Work:Madhouse/MadhouseConfigEd
You don't have to copy any catalogs, because "English" is the built-in language of Madhouse.

1. Start the program.
   This should be easy. Double-click on the icon.
   If you don't have OS 2.0, Madhouse is not the only reason to upgrade. In other words: OS 2.0 is needed. And a BIT of memory. Not for the main program but for many of the blankers the amos.library is required. It is submitted with Madhouse.
2. A window will be opened.
   If not, everything is correct. If you want to open the window, then
   - you DON'T have to press a hot-key which you always forget, and
   - you DON'T have to start the Exchange-Program what you always delete.
   Have you ever used the Tools-menu? Toolmanager is not the only program which is able to merge an item to the list... So select "Madhouse" in the Tools-menu! Madhouse only calls the ConfigEd (which opens the window) without the command of the user, if it cannot find its ENV:Madhouse.prefs.
   If you get a requester which tells you something about incorrect pathes, follow the instructions to quit Madhouse (by running it again after closing the requester), correct the path in the tooltype (see section 0.) and start it again.
3. Now enter the path of the blanker-modules!
   In the directory where you had installed to docs, there is a sub-directory named "blankers". Click on the gadget with the little image in it, next to the string-gadget. The ASL-directory-requester appears. Then select the blankers-drawer. After that, you should see names like "FlyingToasters" and so on. Be sure to select a path which begins with the name of the disk - "Blankers/" is wrong, but "Work:Madhouse/Blankers" will do fine.
   If your directory was the right directory, the gadgets "Use", "Cancel" and some others are enabled.

- If you haven't got a hard-drive...
  Please select "Buffering" in Advanced-Options. By default, "Buffering" is already on. The gadget is in the Advanced-Options-Window or (if you have MUI installed) on the Advanced-Page.
  But please read All about Buffering.
- If you have a hard-disk...

You can deselect "Buffering" in the Advanced-Options-Window (or on the Advanced-Page). But it will be explained later.

4. Now we want to see the blank-Modules!
   Without MUI:
   Click on the cycle-gadget named "Edit" until you see the text "Prefs...". With this cycle-gadget you decide what happens when you click on the list. In this mode, you can edit the preferences of the module which you can select below. (In the preferences-editor is a button which starts the blanker immediately, that's why we use this option.) Now click on a name in the list.
   With MUI:
   Go to the Blankers-Page. The list on this page allows you to open the BlankerPrefs-Window. Double-click on a name on the list.
   With and without MUI:
   A window opens. This is the BlankerPrefs-window. Please click on "Test". Great, isn't it? To abort, press a key or a mousebutton. Now change the parameters with the gadgets above. Click on on "Cancel" to exit and on "OK" to exit and save. The "Duration"-Slider will be explained in the reference-part/the BlankerPrefs-window.
   Choose other blankers in the list and have fun....
   (twenty minuates later) ...finished? If you like Madhouse take a closer look at the other gadgets in the mainwindow, otherwise (?!) click on remove.
5. More information about the list(s)
   Without MUI:
   The list is used for two functions:
   - You can open the BlankerPrefs-window and edit the settings for the blanker-module you clicked on.
   - You can decide which blanker(s) will be used when the time you selected is over.

   You know already how to edit the blanker-preferences: With "Prefs..." in the Edit-gadget.
   If you want to select the blankers which will be used when time's up, select "Selection" in the Edit-gadget. Then the names will scroll two chars, and you see "» " or " " in front of the entrys.
   Perhaps you already expect how this funktion works: with clicks on the names you can toggle the blankers on or off.
   With MUI:
   The MUI-part of the MadhouseConfigEd offers two lists on two pages. The list on the Blankers-Page opens the BlankerPrefs-window, as you saw it. The list on the System-Page lets you decide which blankers should be used when the time you selected is over. By default, all blankers are selected, but you can change the status of every blanker by clicking on its name. The background color of the list changes as well.
6. Select the time after the blanker should start to blank!
   With the "Time"-gadget.
7. Save your configuration.
   Nothing easier than that: Click on "Save". The configuration will be saved as "ENV:"- and "ENVARC:Madhouse.prefs".
8. If you have no hard disk or if you need a password,
   Click here:The Advanced Options

1.3 The Advanced Options window / The Advanced-page

The Advanced Options-window/page offers some additional Features of Madhouse:

1. Passwords
   If you want to use a password, check the "Password/Use"-Button. Enter your password in the string-gadget. BE CAREFUL: Write it under your mouse or on a disk or behind your monitor, but without knowing your password, there will be no chance to go back to your application. (I hope so...)
   After setting the password, Madhouse lets you enter it again to see if you are able to remember your password, and if all used chars are valid for a password (you cannot use <Alt>, <Ctrl> and the Amiga-keys).

   After you stopped a blanker, the password-screen will appear. You see only a string-gadget (which is, in fact, not a real one). You don't see a curser, and all chars you enter will be only displayed as a "*". You have three chances to type the correct password. You can abort the process by pressing Escape or confirm the password with Return. Backspace deletes the right char. Other keys are not supported. Case is sensitive.

   Please check carefully if the Password-Screen is _really_ safe on your computer-configuration. After a wrong password, it can take Madhouse a short time to start the next blanker.

2. Options for floppy disk users
   If you havn't got a hard disk, Madhouse is (at my knowledge) the ONLY modular screen blanker you can use. What do you think will happen, if you have installed Madhouse on a disk an the disk is not in the drive when it's time to blank... All other screen blanker want to attack the disk with their read-processes without checking if the disk is in the drive... And if not, you see a requester like "Please insert disk xy in any drive". But if you are away? (...)

   So if you have no hard-disk, select buffering. If buffering is selected, Madhouse will do these things:
   - If you close the main window or in the moment you run Madhouse, it will copy one of the selected blankers to it's subdirectory in the Ram-Disk.
   - If you go away, Madhouse will start the blanker from the Ram-Disk.
   - Perhaps the disk is available although you selected buffering. Then Madhouse will change the blanker, otherwise always the same will be displayed. (Of course the blankers will only change during the blank-period, if you selected "Change Blanker" in the main window.)
   - If you abort the blanker by clicking with the mouse or pressing a key on the keyboard, will wait for you to insert the disk with the blankers. Of course not with the OS-Insert-Disk-requester, but if you want, you can see a small window on the WB. (If you want this window, select "Ask for Disk".) If you insert the disk, Madhouse will copy a "fresh" blanker in the subdirectory. Otherwise, when you go away the second time, Madhouse will use the old blanker.
   - The little window has also a close-gadget. Use it if you want Madhouse NOT to load a blanker if the disk is in the drive.

3. The black background
   Especially with disks it can take some seconds to load a blanker.

If you don't want to see the Workbench between the blankers, check "open Background/use". Then a black screen will be opened in front of the others while Madhouse is blanking or loading blankers.
If you select also "display Info-texts", you will see the texts "Madhouse: Loading blanker." and "Working. Please wait..." on this screen, with a little logo an the left side.

4. Other options
The first "other option" is the "CPU active"-setting. With this cycle-gadget you can select what Madhouse should do if your computer has something to work on. To start a complex screen-blanker would slow down the calculations. So Madhouse checks the CPU-activity and - if your processor is working - uses this setting to decide what to do. The cycle-gadget has three positions:
- "use all Blankers" means that Madhouse doesn't take care of your CPU at all.
- "only simple ones" influences the random-function of Madhouse. If this option is active, Madhouse will only start Blankers which hardly need the CPU. If you selected only Blankers which use your CPU a lot, you will see a blank screen and after that a message which tells you about this situation.
- "show nothing" tells Madhouse that it has to open a black screen if your computer is working.
If you want to know which blankers are the big CPU-users, see all Blankers.

The second button is the Blanker-Pri-Cyclegadget. With this gadget you can set (as the name says...) the task-priority of the Blankers. If you use "CPU active: use all Blankers", this setting should be "0". (Otherwise, a calculating program will be very slow if Madhouser starts a Blanker which uses the CPU a lot.)
But if you use other "CPU active:" settings, you can set Blanker-Pri to 1. This would be the good combination. If you want your calculations (if there are some) as fast as possible, you can use a Blanker-Pri of 0. Remember that Madhouse checks the CPU before starting a blanker. If a program starts to calculate afters this check, maybe a CPU-using blanker was started and slows down the other process... So decide carefully what you set here!

Finally, there is the third cycle-gadget "Sound" in this category. It will be only available for users who have got MUI.
This gadget switches between the two ways of playing mods while blanking:
- with the medplayer.lirary by Teijo Kinnunen, which just plays MED or
- via Delitracker. Madhouse controls DeliTracker (© Delirium Softdesign, Peter Kunath and Frank Riffel) with its ARexx-Port. You don't have to start RexxMast for this, RexxMast is needed if you want to run ARexx-Macros, but not for the communication of two "real" programs.

Let me say it in this chapter again: you can use this sound-option only if you have got a hard-disk!

1.4 All our blankers!

All blankers are listed in alphabetical order below, with all the information belonging to them.

CrazyPixel
DigitalClock
Drops
Fireworks
FlyingToasters
Glitter
Memory
Note
Shuffle
Skyline
Soccer
SoftwareFailure
Stars
Thunder
Waves
Worldtime

Note that the Duration-Slider and the gadgets below are explained
in Reference/BlankerPrefs-window.

1.5 The blankers - CrazyPixel

CrazyPixel

Created by Aicke Schulz, Version 1.

This effect shows you a bouncing point or a "Wusel" (cannot be translated
into english...).

Mode: switches between both modes. If Random is selected,
the program switches itself.
Wusel lenght: sets the lenght of the Wusel.
Sound: adds sound to the bouncing point.

CPU-usage: Low. Memory-ussage: 160 kB. Executable-size: 25 kB.

1.6 The blankers - DigitalClock

DigitalClock

Created by Aicke Schulz, Version 1.

Like other blankers, Madhouse has a "clock"-module. This one has a very
nice digital-like font, vertical scrolling, a date and an alarm-function.
Note: This effect needs a real-time-clock inside your Amiga.

Countdown: If this sliders is not set to "0", it sets the number of
minutes after the blanker starts to get on your nerves
with a very nice bell-sound.
Language: Is needed to select a format for the date and to switch
between "alarm on" and "Alarm an".

```
          Date: switches the date on or off.
        Scroll: moves the clock.

CPU-usage: Low.    Memory-usage: 180 kB.   Executable-size: 34 kB.
```

1.7 The blankers - Drops

```
                              Drops

                  Created by Aicke Schulz, Version 1.

Drops are running down your screen.

       Number: sets the number of drops which should be displayed
               onscreen.
        Color: Whow, the mega-mix of 5 color-palettes: Ocean, Wine,
               Grass, Caramel and Fresh. Select Random and the
               blanker will choose itself.

CPU-usage: Low.    Memory-usage: 180 kB.   Executable-size: 17 kB.
```

1.8 The blankers - Fireworks

```
                             Fireworks

                   By Carsten Jahn, Version 1.

This blanker shows a nice and colorful firework on your screen. It has
four effects. You can use the sliders to set how often one effect should
appear. If you don't use too high values, FireWorks is able to change
the colors while one register is unused.

          Spirals: sets how often Spirals are used.
             Jets: sets how often Jets are used.
            Balls: sets how often Balls are used.
       Color-Jets: sets how often Multi-Color-Jets are used.
       Pixelspeed: Perhaps you've noticed that this effect becomes slower
                   and faster all the time. Try different values with
                   the Pixelspeed-slider to get the right timing for your
                   machine.

CPU-usage: High.    Memory-usage: 110 kB.   Executable-size: 17 kB.
```

1.9 The blankers - FlyingToasters

```
                           FlyingToasters

                 Created by Carsten Jahn, Version 1.

Yeah, they are back. Being heavily inspirated from the AfterDark-Toasters
```

(not the SuperDark-Toasters), I painted the graphics with PersonalPaint.
They need a 16-color-hires-interlaced-screen and have 6 anim-phases.
Perhaps I'll add a check for the "DblPal"-modus, but I think graphics
don't flicker too much.

```
        Toasters: sets the amount of toasters on the screen.
          Toasts: sets the amount of toasts on the screen. The toasters
                  will "overdrive" them, but not crash them.
           Speed: can be a value from 1 to 7. 7 is the fastest mode.
             Jam: adds jam to the toasts. The perfect breakfast!
Double Buffering: enables, as the name says, double buffering. Normally,
                  the blanker draws only on one screen, and you can
                  perhaps see the "Blitter" working (depends on your
                  Amiga and the settings). With double buffering, the
                  program will draw on one screen (which you can't see)
                  and shows another, older screen. Then it will swap them.
                  Double buffering needs more memory. If it is not
                  available the Toasters won't use double buffering.
   In/Out Moving: With this option switched off, the Blanker will put and
                  erase the Toasters immediately on the program's start /
                  end. If you use "In/Out Moving", the Toasters will flew
                  into the empty screen on startup. Before the Duration-
                  time is over, the Blanker won't put any more Toasts, so
                  that the screen will be left empty again.
                  The last thing is only possible if "Change Blanker" was
                  checked.
```

CPU-usage: High. Memory-usage: 297 kB. Executable-size: 29 kB.

1.10 The blankers - Glitter

Glitter

Created by Aicke Schulz, Version 1.

It's not the last program-technical invention, but a nice effect that
shows ... shows what? Glittering stars.

```
       Number: sets the number of stars. This will slow down the startup-
               process, but not the animation.
   Cyclespeed: sets the speed of the "glittering".
```

CPU-usage: Low. Memory-usage: 218 kB. Executable-size: 15 kB.

1.11 The blankers - Memory

Memory

Created by Aicke Schulz, Version 1.

This is a very good possibility to see what is in your memory. The
blanker shows the first MegaByte of Chipmem. You will see the the

bitplanes of the opened screens, your Workench, icons and windows.
The rest can be seen, but you won't recognize it: program code, texts,
sounds. I once even saw disk-activitations, and a heard that the Blitter
encrypts the data from disk. So you can see *perhaps* even this. I am
not sure. In the top of the screen there is a red line. It shows the
position of the view relatively to the 1 MB chipmem.

Speed: sets the speed of the memoryscrolling.

CPU-usage: Low. Memory-usage: 180 kB. Executable-size: 14 kB.

1.12 The blankers - Note

Note

Created by Aicke Schulz, Version 1.

Note has two functions in one blanker: on the one hand, you can tell the
people who want to see you (while you are not there) where you are or
something else. On the other hand, you can give these people the possi-
bility to tell YOU something. Therefore, they can enter a bit of text.
Note has three backgrounds for the memo pad.

Paper: This switch selects one of the background. By the way,
"Against AIDS" is a real memo pad which you can get in
Germany... see the german doc for the address.
Mode: Mode lets you decide between the sender- and receiver-mode.
If you select "Give Message", you have to use the text-
fields on the right side to enter the text. Every line must
end with <Return>! "Get Message" allows the visitors to
enter a message. If Change Blanker is not active, or if the
blanker is still working when you come back, you can read
the message. If the Duration-time of Note was up and another
blanker was started, you can read the message in the error-
list after stopping the blanker.
Static: With "Static" you can set the time (in seconds) after which
the pad moves again.
Textlines: If you selected Mode = Give Message, you have to use these
gadgets to edit your message. Not that you have to terminate
every line with <Return>!

Error: If the mode is "get message", and the blanker stopped itself to
allow Madhouse to start the next Blanker (Exchange-blanker-option), you'll
get the message as an error.

CPU-usage: Low. Memory-usage: 210 kB. Executable-size: 48 kB.

1.13 The blankers - Shuffle

Shuffle

Created by Carsten Jahn, Version 1.

Have you ever seen a modulary screen-blanker without a "Shuffle"-module?
Here is our one. But I didn't wanted to write such a simple Shuffle, but
one with some new features.
So this Shuffle is able to restore a puzzle.
It is AGA-Chipset-sensitive and recognizes other Screenmodes like DblPal
and others. You can use more colors, too. It will turn off the 3D-Grid
if the colors are "wrong". And I hope it will run on graphic cards which
use Screenmodes (tell me about it, I don't have such a card).

```
         Mode: sets the Shuffle-mode. "Shuffle" is the normal Shuffle as
               you surely know it. The other modes can only be used with
               "Change Blanker" (main-window) activated, because the
               blanker has to know where to stop. "Shuffle & Restore"
               first shuffles the screen, and then solves the puzzle.
               "Restore" starts with a shuffled screen and then restores
               it in the "Duration"-time.
        Speed: sets the speed of the shuffling.
         Grid: turns the 3D-Grid on or off.
```

Errors: perhaps a text like 'The "Shuffle & Restore" mode and the "Restore"
mode can be only used if "Exchange Blanker" is activated.' will occur.
This means that you tried to use a Resore-mode although the blanker cannot
know when to finish, but he has to know that because he does not know what
to do after the restore is finished.

CPU-usage: Low. Memory-usage: ??? kB. Executable-size: 13 kB.
(The memory-usage depends on the screen copied by Shuffle and the mode-
setting.)

1.14 The blankers - Skyline

Skyline

Created by Aicke Schulz, Version 1

This is Aicke's fantastic skyline-simulator which shows all hillbillies
out there how a real town looks. Pixel for pixel a skyline appears.
Some houses have flashing red lights on their roofs. The moon is moving,
too! After some minutes you can see the city and the different types of
windows.

```
  Moonphase: is used together with "Setting". If you don't select "Random"
             here, you can use "Setting" to set the moonphase.
    Setting: Allows you to define how full the moon is.
  Starcolor: "Off" switches all stars off, "Random" lets the computer de-
             cide and with all other options you can set your favourite
             starcolor.
     Number: The number of stars.
    Y-Limit: This slider defines the vertical limit of the stars. Skyline
             won't set any stars under the screen-line you define here.
        Sky: Different copper-lists for the background. "Night" switches
             the copper-list off, I don't explain Random again...
```

```
CPU-usage: Low.       Memory-usage: 200 kB.     Executable-size: 32 kB.
```

1.15 The blankers - Soccer

```
                                Soccer

  Created by Carsten Jahn, a lot of anims by Aicke Schulz, Version 1.

Hey, have you ever seen somethink like that? A soccer game as a screen
saver. Forget your TV-set, now you can watch soccer while setting in front
of your Amiga. Perhaps, with this blanker, soccer will become more popular
in the U.S.A.
Soccer has no sound, it's your task to cheer.
If there is enough memory, Soccer uses Double Buffering.

 Line-up White / Red: With these switches you can select the line-ups for
                      the two teams.
 Keeper intelligence: Of course, there is no intelligence in your Amiga,
                      but this slider allows you to set the "synthetic"
                      intelligence of the goalkeepers.

CPU-usage: High.       Memory-usage: 378 kB.     Executable-size: 45 kB.
       If there is a lack of memory: 260 kB.
```

1.16 The blankers - SoftwareFailure

```
                          SoftwareFailure

                 Created by Aicke Schulz, Version 1.

After a short pause you come back into the room where your computer
stands. And what do you see there: a red, flashing box around a short
text. Ah, no! What was the reason for this crash? But don't mind, after
you've pressed the left mouse button you can work on. SoftwareFailure
tries to shock nervous poeple...

     Effekt: SoftwareFailure offers three effects to you, these effects
             start after a time which you can set with "Wait".
             "Bounce" makes your Guru bounce, "Crazy" makes the letters
             move and "Melt" does something else.
       Wait: sets the time (in seconds) after one of the effects starts.

CPU-usage: Low.        Memory-usage: 150 kB.     Executable-size: 29 kB.
```

1.17 The blankers - Stars

```
                                Stars

                    Created by Carsten Jahn, Version 2
```

What should I say... This is the Madhouse-version of the well-known star-
blankers. They have an new effect that I haven't yet seen on an Amiga:
The turns and the backwards gear. Try it out.
Many settings....

```
 Number of Stars: sets the number of stars.

Normal movements (movements ON the z-achsis):

         Maximum: speed of the stars.
         Manimum: speed of the stars. Negative values = bachwards gear.
         Changes: sets how often the stars change their speed (between the
                  two values you can select with Minimum...Maximum.
           Start: sets the start speed.

Turns (movements AROUND the z-achsis):

         Maximum: speed of the turn.
    Acceleration: sets how fast the stars get their new speed.
         Changes: sets how often the stars change their turn-speed.
           Start: sets the start speed.

Shifts (movements on the x- and y-achis):

           Speed: sets the speed of the shiftings.
           Usage: How often this effect will be used. "0" means never,
                  "10" means always.
             X/Y: Which achis (or both?) will be shifted.

CPU-usage: High.       Memory-usage: 70 kB.     Executable-size: 11 kB.
```

1.18 The blankers - Thunder

```
                              Thunder

                Created by Aicke Schulz, Version 1.
```

Thunders inside your Amiga! With the realistic sound. As you know, sounds
are slower then light. So you can hear a thunder always some seconds after
you saw him. These "some seconds" change with the distance of the thunder.
This was implemented in Thunder.

```
 Flash Interval: with this button you can choose how long "Thunder" will
                 wait between two thunders.
       Distance: This value tells "Thunder" how long it has to wait
                 between the thunder and the sound.
          Sound: switches the sound on or off.

CPU-usage: Low.       Memory-usage: 185 kB.     Executable-size: 32 kB.
```

1.19 The blankers - Waves

Waves

By Carsten Jahn, many colorsets by Aicke Schulz; Version 1.

Here is a blanker which mades very interesting Plasma-effects. I don't want to explain in detail how this was made (because I had even problems explaining it in german, sorry).
On the left side of the BlankerPrefs-Window you see a selection of color-palettes. Waves uses one of these which have are activated (check-mark). If you deactivate all colorsets, Waves will choose itself.
On the right side are the parameters which influence the shape of the waves:

WaveSize: influences the thickness of the waves.
XWaves: sets the width of the waves.
YWaves: sets the height of the waves.
XSize: is the maximum of the horizontal sinus-arc.
YSize: is the maximum of the vertical sinus-arc.
XSpeed: sets the speed of the horizontal screen-scrolling.
YSpeed: sets the speed of the vertical screen-scrolling.

It is useful to experiment with the parameters.

CPU-usage: Low. Memory-usage: 212 kB. Executable-size: 14 kB.

1.20 The blankers - Worldtime

Worldtime

Created by Aicke Schulz, Version 1.

Worldtime shows you a map of the world and a lot of cities (red points). Now it draws little boxes near the cities, including the name of the city and the time there.

Order: decide here if you want to see the cities in alphabetical order or in no order ("Random").
Wait: How long the box will stay near a city (in seconds).
Hours and Minutes: I hope your Amiga-clock is set correct. This parameter says the blanker where you live.
Please set it to the time-lag of the place where you live, compared with the MEZ (Mitteleuropäische Zeitzone, Germany for example needs the values "0" and "0"). Normally, you'll just have to change the hours.
Language: Sets the format of the date.

CPU-usage: Low. Memory-usage: 290 kB. Executable-size: 51 kB.

1.21 The error system

Madhouse knows a lot of errors... The errors are devided into two groups.

One the one hand, the main-program can display errors. You see them in a window on the Workbench, with a "Continue"-gadget below.

On the other hand, the blankers check their environment and the selected options. If something goes wrong, they don't display the error themselves, but they give the text to Madhouse. If you started the blanker yourself in the BlankerPrefs-window, the window will get a bit larger, and you can read the error in the bottom of the window.
If the blanker was started by Madhouse, Madhouse will first try to start other blankers you selected. If no blanker is left, you will see a black screen. You can close the screen like you quit the blankers: by pressing a key or a mousebutton. Now you see the Workbench and the Errorlist, a window which shows you all the errors which occured. You can scroll through it and perhaps change some settings afterwards.

If you want more information about a blanker-error you can look at it's description.

The errors from the main-program are explained below (excluding these ones which have to do with "not enough memory"-situations).

PROBLEMS WITH FILES AND DIRECTORYS
Generally: Not all files are important, the files including settings and configuations can be restored by simply setting these settings again and saving them. (Files: ENV:Madhouse.prefs and blankers/.../prefs)
All other files are difficult to restore, like the blanker-executables and ther gadget-files.

File ENV:Madhouse.prefs does not exist.
This is not a real problem, it simply shows you that you have to configurate Madhouse again, because it's config-file "ENV:Madhouse.prefs" is not available.

Please select a COMPLETE path,...
Why a complete path? A path like "/blankers" or "df0:Madhouse/blankers" is not enough, because Madhouse has to identify the Volume containing it's blankers, if you use the buffering-mode. So please select a path like "Work:Tools/Madhouse/blankers".

BlankerInfo-Chunk does not exist!
In the "gadget"-file of this blanker is no BlankerInfo-Chunk. So Madhouse cannot read wether the blanker uses the CPU a lot or not.

Unknown macro: xx
In the gadget-file of the called blanker is a macro xx which doesn't exists.

Couldn't get a lock on this directory!
Perhaps you entered a path in the ASL-directory-requester which isn't correct, or a disk-copy-program locked the whole disk.

Couldn't create the subdirectory "Ram:..."
Perhaps your Ram:-Disk isn't mounted.
(By the way: If you want Madhouse to use another Device for it's storage-directory, you can only use a binary editor like "AZap" (© by Denis

Gounelle). But I think this won't be necessary, because a device like
DF0: won't be fast enough, if you want to use a statical RAM-Disk like
VD0: or SD0:, you have to delete the storage-directory after every reset
(otherwise you will get this error, because Madhouse is NOT resident), and
to use a hard-disk for that is just joke, the directory includes only
very small files (except the blankers, but with a hard-disk you can turn
the buffering-mode off).

  You started Madhouse the second time. Do you want to remove it?
To start Madhouse twice is another possibility to quit it. This requester
asks you if it is that what you want.

  No amos.library
Madhouse wants to copy the file LIBS:amos.library to the RAM:-Disk, if
"Buffering" is on. You have not installed the library.

A FILE IN THE BLANKER'S SUBDIR IS MISSING!
Generally: See "generally" above.

  Couldn't load the "gadget"-file!
The gadget-definition is needed to open the window.

  The "prefs"-file is not in the subdir!
Not a real problem. You will have to configurate this blanker again.
Move every slider, on the startup without a prefs-file in the blanker's
subdir every slider has the value 0.

  Couldn't access "RAM:Madhouse_Storage/prefs"!
There is no prefs-setting-file in the blanker's subdirectory.

  Couldn't start this blanker:
The "blanker"-file in the blanker's subdir is missing.

THE OTHER STUPID ERRORS...
Perhaps all errors are stupid, but here are the real stupid one's.

  Problems with your password
You cannot use <Alt>, <Ctrl>, <Caps Lock> and other keys while
entering the password. So your password mustn't have capital letters.
You can use the numbers, öäü, ß.

  Need a stack minimum of 4.096!
The stack is just a block of memory which is allocated by DOS (not by the
program) for every started program. So the program itself has no ability
(on my knowledge) to size this stack. The user who starts the program has
to size this stack. But don't worry: normally it's easy and you don't have
to do it often. Where you input the stack size (which
should be 4096 Bytes) depends on where you start Madhouse. I explain
three ways here:
1. From the Workbench or by moving Madhouse's icon into the WBStartup-
   drawer:
   Click once on Madhouse's icon. Select "Information" in the "Icon"-Menu.
   A requester appears. Click in the box near "Stack:" and correct the
   value.
2. From the Shell or the S:User-Startup / S:Startup-Sequence:

Every Shell-window has it's own stack-value. The programs you run from this Shell get this value. You set this value by typing
stack 4096 <Return>
in this window before starting Madhouse.
In the Startup-Sequences you insert this line before the Madhouse-call.
3. From the Toolmanager:
Enter the value 4096 in the program-requester for Madhouse.
4096 bytes is the normal value for Amiga-programs - so there shouldn't be any problems, but who knows?

208 Bug #1 in program!
It's hardly possible to get this error. The Bug #1-error is only a help for me to write a program with less/no bugs.

No chance to run Madhouse on your 1.3-machine!
Arrrrgh! OS 2.0 is needed by so many programs, why do you use this terrible 1.x-version? You can get the new ROM's and the software in a lot of Amiga-shops, and if your old software needs 1.3 you can buy a switch for two ROMs. It's very easy to insert the new ROM's into EVERY Amiga, and it doesn't cost much!
If you have OS 2.0 and you get this message, you have a real problem...

Couldn't open asl.library V37!
The asl.library is not in the LIBS: path.

Couldn't generate AppItem!
Couldn't create Message-Port for AppItem!
Internal errors.

Can't read the "gadget"-file of this blanker! (You need a newer Version of Madhouse)
The first line "Madhouse, Blankerwindow-Def V1" is not in the prefs file of this blanker! Perhaps you made a fault in creating a gadget-file, or we made a newer Madhouse which is _that_ new that it is not able to load the old files (I think I'm not going to change something in the file structure, so all old gadget-files will be compatible to new ones.)

File mismatch: Chunk "Dimensions" must be the first Chunk!
This Chunk MUST BE the first Chunk of the prefs-file.

BlankerPrefs-window too big for this screen!
Try a bigger Workbench or a lower Width/Height-Value.

Enter a higher value in "CHUNK:DIMENSIONS" - Gadgets!
Don't think about it... do it!

Wrong Statements in CHUNK:MUI-PREFSORDER.
As the message says, you made a mistake in that chunk.

Couldn't access the Config-Editor.
Madhouse wasn't able to start the programm "MadhouseConfigEd". If it was able to get the settings, it can continue running, and you can change the ToolType. If the settings were not available, Madhouse has to quit.
The previous requester and this one tell about this situation.
So quit Madhouse (if you need to do so) and change the Tooltype "CONFIGED=" of Madhouse, so that it includes the complete path and the name

of MadhouseConfigEd. If you do not want to start Madhouse with WBStartup
or with the Workbench, but with the Shell, remember that Madhouse searches
the ConfigEd in the current directory in this case.

  The MadhouseConfigEd can only be started by Madhouse itself.
As the requester sais, MadhouseConfigEd can only be started by Madhouse,
not by the user. (This was necessary, because both programs use the
Madhouse_Storage-drawer, which can result in conflicts if both programs
run at the same time.)

1.22 All about the buffering-option!

  Madhouse offers a lot of intern functions which help users who don't
have a hard-disk. You enable all these functions by checking "buffering"
in the Advanced-Options window/page.

If this option is enabled, Madhouse will
- permanently store a blanker in the "RAM:Madhouse_Storage/"-Directory.
- load a new blanker in this directory, if you selected more than one
  and the blanker was used already. If you selected "Ask for Disk",
  Madhouse will open a new window which shows that Madhouse is waiting
  for it's disk.
- copy the libs:amos.library to the RAM:-Disk, so the blankers can reach
  it.

  If a blanker allows you to configurate a filename, and the blanker wants
to load it, it should be in the blanker's subdirectory. When Madhouse
copies a blanker to the RAM:-Disk, it copies all files in it's sub-
directory with him. Then you can just enter "File_xy" and the blanker
will try "RAM:Madhouse_Storage/File_xy" or
"Work:Madhouse/Blankers/NiceBlanker/File_xy"; depending on what Madhouse
tolds him. But at the moment no blanker uses extern files.

1.23 The reference-part

  In this part of the Madhouse-documentation, you can look up the functions
of all the gadgets and some other facts, without reading the long work-
shop.

Without MUI:

The main window
The BlankerPrefs-window
The Advanced Options window
The Error List
The Ask-For-Disk-window

With MUI:

The main window
The BlankerPrefs-window

```
                    The Error List
                The Ask-For-Disk-window


Tooltypes

                         CONFIGED
                         QUIETQUIT
```

1.24 Reference: Tooltypes - CONFIGED

```
Type the complete path of MadhouseConfigEd after the "=", example:

CONFIGED=Work:Madhouse/MadhouseConfigEd

Do not use spaces next to the "=".
```

1.25 Reference: Tooltypes - QUIETQUIT

```
If this tooltype is set, Madhouse does not bother you with a requester,
if you wanted quit it by starting it again. Remove it or set it in
brackets to enable the requester.
```

1.26 Reference: The main window

```
I. How to open this window
You can open the window by selecting the item "Madhouse" in your Workbenchs
"Tools"-Menu. This menu is only available, if you activate a Workbench-
window or if you click on the Workbench-screen.

II. The gadgets
Click on the gadgets to get information about them.

                      Edit    Prefs...
                          The List
                      Path Work:Tools/Ma
                       Advanced Options
                      Time    |
                          Change Blanker
                             Save
                              Use
                            Remove
                              Info

III. How to save the settings
Just click on "Save".

IV. How to close the window
Click on "Save" or "Use" to close the window. Remember: while this window
is open, Madhouse won't check if it's time to start a blanker.
```

1.27 Reference - Main window: Edit

```
This gadget allows you to change the contents of the list below.
It can display two texts:
   "Selection": Now you can use the list to disable/enable the blankers.
   "Prefs...":  In this mode, you can open the BlankerPrefs-window to
                change the setting for the blanker you selected in the

If the gadget is disabled, a wrong path for the blankers is
selected. In this case, use the Path-gadget to select a better path.

See also: the workshop, point 5 (the list and the Edit-gadget).
```

1.28 Reference - main window: The list

```
The list is connected with the Edit-gadget. Select with the Edit-gadget
what you want to edit, and edit it with this list.
If you want to edit the preferences of a special blanker, select "Prefs..."
in the Edit-gadget and click on the blanker in the list.
To select the blankers which will be used later, select "Selection" in the
Edit-gadget and then the blankers in the list.
A selected blanker has a "» " in front of his name.

If the list is disabled, a wrong path for the blankers is selected.
In this case, use the Path-gadget to select a better path.

It may be that the list contains the old data after you have changed
something in the blankers-directory. Then just click in the Path-Gadget
and type <return> to read the new path. (Madhouse doesn't read the
directory on every startup. It would use too much time while booting.
And, of course, you could change the directory while Madhouse is running.)

See also: the workshop, point 5 (the list and the
Edit-gadget).
```

1.29 Reference - main window: Path (Stringgadget)

```
If the Use-gadget and some others are disabled, use this gadget to enter
a correct path for the blankers.
Be sure to insert a complete path, with the name of the volume in it.
Madhouse needs this name for the Buffering-option.

The path is correct if it ends with "blankers" (if you haven't changed
the directoryname). Madhouse recognizes the path if it has the file
"the_right_drawer" in it, so don't delete this file.
```

1.30 Reference - main window: Advanced Options

Madhouse has even more options. Select this gadget to get a new window
with new gadgets and new options...
See Advanced Options for the description of the new
gadgets.

1.31 Reference - main window: Time

If you leave your Amiga, Madhouse waits a special time before its starts
the first blanker. Select this time with the Time-gadget (in seconds).

1.32 Reference - main window: Exchange Blankers

With this gadget you can specify if Madhouse changes the blankers after
a while. If you turn it on, the "Duration"-Sliders in the BlankerPrefs-
windows will be enabled. With the "Duration"-Slider of every Blanker-module
you can select how long it will work.
Of course it is not very intelligent to change the blanker if only one
is activated, in this case the "Exchange Blankers"-gadget will be disabled.

1.33 Reference - main window: Save

Exit window and save options. This will be saved to
"ENV:" resp. "ENVARC:Madhouse.config":
- The options from the main window / system page.
- The options from the Advanced-Options-window/page.
- The current position of the "Waiting for Disk..."-Window.
- The Duration-setting of every blanker.
- The names of the blankers and if they are selected or not.

So if you work on the blankers-drawer, Madhouse won’t recognize it until
you load the directory again (by pressing <return> in the "Path"-Stringgad-
get). On the other hand, Madhouse can just read all the information out
of it’s config-file, without having to read the directory on every startup.

1.34 Reference - main window: Use

This gadget closes the window without saving all the stuff.
You can open it again by selecting "Madhouse" in the WB’s Tools-Menu.

1.35 Reference - main window: Remove

Removes Madhouse.

1.36 Reference - main window: Info

A window will appear which shows some information about our program.

1.37 Reference: The AskForDisk-window

The AskForDisk-window (the one with the little disk in it, appearing only if "Ask for Disk" is selected) shows you that Madhouse needs your Blanker-disk to be able to start a new blanker the next time.
If you close the window, Madhouse won't load a new blanker even if you insert the disk.

1.38 Reference: The BlankerPrefs-window

I. How to open this window
To open the BlankerPrefs-window, you have to open the main window first. Then select "Prefs..." in the Edit-Cyclegadget and click on a blanker.

II. The gadgets
Based on which blanker you selected, the gadgets in this window are differend. But four gadgets can be found in every BlankerPrefs-window:

II.1 The Okay-gadget
Saves the preferences you set above to "blankers/blankername/prefs".
Notice: You have to use the Save-gadget in the main window to save the Duration-Value!

II.2 The Test-gadget
This gadget runs the blanker with the selected options.

II.3 The Cancel-gadget
Quits the Preferences-Editor without saving the prefs.

II.4 The Duration-slider (MUI: not here, but on the Blankers-page)
You can select how long this blanker will be shown if Exchange Blanker in the main window is selected. Otherwise, this gadget is disabled and wouldn't make much sense.

1.39 Reference: The MUI-window

The gadgets on the bottom of the window:
Save
Use
Remove

System-page
The List
Path

```
                    Time
                Exchange Blankers

                  Advanced-page
        See       Advanced Options

                 Blankers-page
                   The List
                   Duration
                    Author
                   CPU load
                   Version
            The Sound-Menu as a PopUp

                    Info-page
                  Textgadget
```

1.40 Reference - main window/System: The list

The list makes it possible to select/deselect the blankers. Only selected blankers will be used for blanking. Using this list can be uncomfortable if you run MUI with the wrong settings: this is the case if all blankers are deselected if you just click into the list.
You can prevent MUI from doing so if you select "always" in the "select"-cycle.

If the list shadowed, a wrong path is selected.

1.41 Reference - main window/Blankers: The list

The list on Blankers-page has two tasks, depending on how you click:

- one click:
will display the information of this blanker in the right side of the window. The duration-gadget will be updated, too.

- doubleclick:
The BlankerPrefs-window will be opened. If the selected blanker has no information to build a MUI-window, a normal one will be opened.

1.42 Reference - main window/Blankers: Duration

With this slider, you can select how long the highlighted blanker will work, after tis time (in minutes) Madhouse will start another blanker. Note that this will only happen if "Exchange Blanker" is selected. Otherwise, you cannot use the gadget.

1.43 Reference - main window/Blankers: Author

Here you can see the name of the author of this blanker.

1.44 Reference - main window/Blankers: CPU load

Here you can see if the blanker uses the CPU a lot ("medium to high") or
if he hardly uses it (low). This is necessary for the "CPU active"-setting,
which will not take a "medium to high"-blanker if "use simple ones" is
selected there.
See Advanced Options for further explanation.

1.45 Reference - main window/Blankers: Version

Here you can see the version-number of the selected blanker.

1.46 Reference - main window/Blankers: Sound

With this sound-menu, it is possible to assign a music-module to a blanker
and to hear it.

For this purpose, the medplayer.library or the DeliTracker is used,
depending on what you set on the Advanced-page. If you want to use the
DeliTracker, it has to be running; if you want to use the medplayer.-
library, you need it in LIBS: and you can only use MED-mods.

First of all, you'll have to select a blanker (click one time). If you
already selected a mod for it, you will see it in the string-gadget. Of
course you have not, otherwise you won't read this chapter... So click on
the popup-button next to the string-gadget.
The Sound-Popup appears. It has the following gadgets:

Add...
Use this button (and select a filename) to add a file to the soundlist.

Del
Deletes a file from the soundlist. Every blanker which had this mod in his
sound-setting has now no module any more.

Tapedeck-play-arrow
Do you really need an explanation for this gadget? Arrrgh, here it is:
this gadget plays the selected module.

Tapedeck-stop-button
Stops the music.

The soundlist
The task of the soundlist is to handle the modules you want to use with
Madhouse. You can double-click on the list to use the selected mod with

the selected blanker and to close the popup. But the real intention for me to program this list was the fact that every user expects a list when he or she clicks on an popup-button...

By the way, you can enter the pathname directly into the string-gadget, but that's boring, isn't it?

If you think that Madhouse saves the list itself (your hard work!), you're wrong. Madhouse only saves the blanker-sound-settings and will rebuild the list on the next start of the program. So if you add a module to the list and if you don't use it, it will be erased after the MadhouseConfigEd quits.

Now here is a nice hint from Aicke: if you want the Delitracker to be quiet after Madhouse-activities, disable "Play at Start" in the Delitracker-Config.

Stop! Please read this section, too! It is very important that the sound-feature of Madhouse can only be used with a hard-disk. But there is a way to use it without a hard-disk, but only with DeliTracker:
Copy the module of your wishes into the subdirectory of the blanker which should use it.
Select Buffering on the Advanced-Page (should be already on...).
Select "RAM:Madhouse_Storage/mod.Module" in the sound-setting. You cannot use the filerequester for that, because the file is not there at the moment. Of course, you'll have to insert the name of your module instead of "mod.module".
See also Buffering-Option.

1.47 Reference - main window/Blankers: Info

In the last page, you can find some information about us, and my hand-drawn-Madhouse-logo!

1.48 Stuff for programmers: Adding own modules to Madhouse!

Madhouse is an "open system" which allows you to add own modules.
The first question should be:
Can I use my programming language?

After this, you can write and compile the blanker.
How should I write my blanker?

Now you can include your blanker into Madhouse's directory-tree.
My own directory!

This is all you have to know if you want to write a very simple blanker. But for a real blanker, you need a BlankerPrefs-window. In fact, you have to write the "gadget"-file. If you want to open the BlankerPrefs-window, Madhouse reads this file and creates a "User-Interface" for your blanker.
The gadget-file

That’s all! If you have problems, please write us!

Now, I want to tell something to people who really want to write a blanker and to distribute it:
1. Please don’t write a blanker like Lines. Of course it’s your decision to distribute a blanker or not. But it would be great if the level of the blankers could be kept. Your idea should be a bit unusual.
2. Perhaps you think of distributing your blanker together with Madhouse. Please don’t do so yourself. This distribution mustn’t be distributed in a modified form.
   If you want to see your blanker among ours, please send it to us. Then it will appear in the next version. But in this case it could be that we don’t want to include your blanker (this seems to be arrogant, but what shall we do if someone really wrote a Lines-blanker?!).
   Don’t be angry if we return your blanker with three pages suggestions for improvement...

If you want to use AMOS as a programing language, there will be some problems if you don’t use the correct compiler-settings. Please look at the listing in the developer-drawer.
C-Programmers should look at the C - listing in the same drawer.

1.49 Can I use my programming language?

Yes.

Perhaps you think it’s not possible with a language like Basic, but it is. Many of the blankers were written in AMOS, others in C++. These very different languages show that Madhouse is compatible to a lot of languages.

You can use every language which
- offers a compiler to create executable files
- has commands to open a screen
- has commands to read and write files.

These languages are:
Assembler, AMOS (buy the compiler!), C, C++, Pascal, Modula, Oberon, GFA-Basic (buy the compiler), Blitz-Basic, Amiga-Basic (buy a compiler), Amiga-E and a lot of languages more.

The only language I know which cannot be used is Arexx. A compiler is available and you can read and write files. But it can’t open screens. (Perhaps there’s an Arexx-Library available which offers you this function, then you can use even Arexx.)

1.50 How should I write my blanker?

To answer to this question, I describe what happens when Madhouse starts a blanker.

Every blanker should have settings. You can set them with the Blanker-Prefs window of the blanker. Madhouse writes them to (for example)

".../blankers/BlankerXY/prefs". When Madhouse starts a blanker, it copies the "prefs"-file from the blanker's directory to the RAM:-Disk ("RAM:Madhouse_Storage/prefs"). Then it adds two lines: How long the blanker can blank ("Duration") and what the directory for it's data is. (This information is only necessary if a blanker wants to load data, the "prefs"-file is always in "RAM:Madhouse_Storage/".
Such a prefs-file can look like this:

```
3
1
RAM:Madhouse-Storage
```

Here, the blanker has only one setting: 3. This "3" can be changed by the user, perhaps with a slider in the Blanker-Prefs window. Or the "3" means that the user selected the fourth (!) entry of a cycle-gadget. How to create the gadgets will be explained later.
The last two lines include always the "Duration"-Time in Minutes (here 1) and the path for other datas. If the blanker has more settings and more gadgets, the prefs-file is longer...

```
25
$A text, string-gadgets are supported, too!
22
0
Madhouse_Disk:Blankers/MyBlanker
```

This blanker has three settings. You see, string-gadgets in the BlankerPrefs-window are possible, too. But you get the text with a "$" in front of it. Here is the duration-value 0, which means that the blanker doesn't have to check when it has to quit.

After reading this file, the blanker can start. He should open a screen and react on the settings. While "blanking", he should test if the user pressed a key on the keyboard or clicked with the mouse. Please don't test if the user moved the mouse. This can happen even if the user doesn't want to stop the blanker. If the "Duration"-value is not 0, the blanker has to check when it has to stop.
If the blanker stops because of a user-action, it has to create the file "RAM:Madhouse/Stopblank" (by just opening (for writing) and closing it). Otherwise, because of the "Duration"-value, the blanker can just quit. The screen must be closed, of course...
Perhaps an error occurres. The the blanker has to APPEND the error into "RAM:Madhouse_Storage/errors".

1.51 My own directory!

As you know, Madhouse has an own directory for every blanker. This has many advantages, its easy to store all files of a blanker in one directory. The directory has another special Aufgabe for Users without a hard-disk: If the blanker should load a file that the user selected, the user can simply copy this file into the blanker's directory and type in the filename without the path. Buffering must be selected, and the blanker can just try to open the file. He should try it twice: With..
1. the filename. This is for users with a hard-disk, who type in the complete filename. Otherwise this attempt will fail. Then the blanker

```
can try
2. the path from the prefs-file (last line) + the filename. This would
be always correct in the case I explained above.

  So create a dubdirectory with the name you want in the blankers/-
directory. Please don't use a too long name, because it has to fit in
the listview-gadget of the main-window.

  Now copy your blanker-executable in it and rename it into "blanker".

  Create an empty file named "prefs" and copy it into your sub-directory.
(It's difficult to create a real empty file [lenght=0] with a text-editor
or such a thing. You can use the prefs-file from the developers-drawer.)

  Then run Madhouse and read the blankers-directory with the Path-string-
gadget. You will see your blanker in the list. You can select/deselect
it, but you cannot open the BlankerPrefs-window. You can run it by
selecting it and deselecting the others, and clicking on "Use". Now
wait....
```

1.52 The gadget-file

```
I hope you know already that "gadgets" are the little boxes, where you can
move the mouse cursor to and click with the left mousebutton...

Now we want to take a look on such a gadget-file.
(Here the gadget-file of CrazyPixel.)


Madhouse v1.0, BlankerPrefs-Window

CHUNK:DIMENSIONS
Gadgets 3
Texts 5
Arrays 1

CHUNK:WINDOW
338,62

CHUNK:DURATION_POS
112,53,150,12


CHUNK:GADGETS

Cycle,112,4,150,12,Mode,TextLeft
1,Done
3,Bouncing Point,Wusel,Random

Slider,112,19,150,12,Wusel lenght,TextLeft
5,SlMin,1,SlMax,20,Done

Checkmark,112,34,26,12,Sound,TextLeft
1,Done
```

```
CHUNK:BEVELBOXES
2
0,0,330,49
0,49,330,20

CHUNK:BLANKERINFO
Aicke Schulz
1
0
5000
```

You can see:
1. The first line includes a text which MUST be in the first line. Madhouse recognises it's gadget-files with this line.
2. The gadget-file is organized in Chunks. This means that every Chunk has a Chunk-Header which identifies it (for example: CHUNK:GADGETS). If the Chunk-Header is unknown, Madhouse skips it.

I'll explain the Chunk-contents later. Very important to know is, that
- you can mix up the Chunks, there is no defined order. BUT: The Chunk "Dimensions" must be the first one!
- you can insert empty lines between two Chunks. But not in the Chunks. Exception: CHUNK:GADGETS allows blank lines between the single gadget-definitions.
- You must write the Chunk-headers and their contents correctly, case is sensitive.

Now the explanation of the Chunks (a "*" means optional, you don't have to write the Chunk in the gadget-file, but you can).

```
  CHUNK:DIMENSIONS
  CHUNK:WINDOW
  CHUNK:DURATION_POS
  CHUNK:GADGETS
  CHUNK:BLANKERINFO
* CHUNK:BEVELBOXES
* CHUNK:TEXTS
* CHUNK:LOCALE
```

Up to now, I said nothing about MUI-windows: if you want a MUI-window for your Blanker, you should read the next chapters. But creating gadget-files which have no chunks for non-MUI-windows won't work.
In the next chapters I explain the usage of two chunks for MUI-windows. As the MUI-syntax differs a lot from the one you know already, I still have a lot of work to do... I hope everybody can understand my explanations, although I will write them very fast (I want to FINISH this english doc). If you don't, please look into the MUI developer kit (available as FD). But I think you will get that. So start the work:

```
MUI for MUI-greenhorns.
    The MUI-groups
    The MUI-gadgets
The CHUNK:MUI-PREFSORDER
```

1.53 CHUNK:DIMENSIONS

```
This very short Chunk must be the first one (was explained above).
It contains three values:

CHUNK:DIMENSIONS
Gadgets x
Texts y
Arrays z

x = The amount of gadgets in your BlankerPrefs-window. Don't count
    the Duration-slider and the three buttons on the bottom of the
    window, Madhouse will add them later.
y = The amount of texts. You can get this value by calculating:
    y = x (Gadgets have one text for their name) + the amount of all
        possibilities of Cycle- and Radio-Gadgets.
    You don't add the number of texts of the Text-Chunk to this value.
    So you just add the amount of gadgets to the amount of Cycle-
    entries, and you have the value.
    If you are not sure, you can enter a higher value here.
z = The amount of Cycle- and Radio-Buttons.
```

1.54 CHUNK:WINDOW

```
CHUNK:WINDOW
x,y

x = The width of your window in pixels.
y = The height of your window in pixels.
```

1.55 CHUNK:DURATION_POS

```
CHUNK:DURATION_POS
x,y,w,h

As explained in CHUNK:DIMENSIONS, you don't count the Duration-slider
and the three buttons "OK", "Test", "Cancel" to YOUR amount of gadgets.
So you cannot set the position of them like you do it with the other
gadgets.
Madhouse spreads the three buttons automatically on the bottom of your
window. But you can set the position of the Duration-slider yourself,
with this Chunk.

x = Left edge of the Duration-slider, relative to the left border of the
    window.
y = Top edge of the Duration-slider, relative to the top border of the
    window.
w = width of the Duration-slider.
h = height of the Duration-slider.
```

1.56 CHUNK:GADGETS

```
The CHUNK:GADGETS contains this:

type,x,y,w,h,name,flags
number of tags,tag1,data1,tag2,data2...
[number of array-elements,element1,element2,...]

for every gadget. This sounds very difficult. But I'll explain this
bit of text for every gadget-type.
Look at the other gadget-files, open their BlankerPrefs-window and
compare!



The first line is always similar:
   type = The type of the gadget. This value decides between Slider-,
          Radio-, Cycle-, String-, Number- or Checkbox-Button.
          You can write here:
          String       for Stringgadgets,
          Checkmark    for Checkmark-gadgets,
          Slider       for Sliders
          Cycle        for Cycles
          Number       for Integer-gadgets or
          Radio        for Radio- (MX-) gadgets.
x,y,w,h = The position and dimensions of the gadget. x = Left edge,
          y = top edge, w = width and h = height.
   name = The name of the gadget
  flags = Allows you to set the position of "name".
          TextLeft     for a text which is on the left side.
          TextRight    for a text which is on the left side.
          TextTop      for a text which is above the gadget.
          TextBottom   for a text which is below the gadget.


The second line for each gadget is the tag-line. Some gadgets need special
information, for example Sliders: they want to know what the minimum and
maximum value is. Other gadgets, like CheckMarks, Cycles and so on doesn't
need tags: for them you just write

1,Done


A Slider{UB} needs tags. For that kind of gadget you write

5,SlMin,minimum,SlMax,maximum,Done

Of course you'll have to insert your values for "minimum" and "maximum".


If you want, a Stringgadget can have tags, too. But here they are
optinional, you don't have to use them. If you want less the 500 chars
as maximum for the stringgadget, you can use

3,StMaxChars,maximum of chars,Done
```

Maximum of chars has to be less then 500. Otherwise, 500 will be used.

You can write the keywords ("String", "StMaxChars", "Done") like you want,
case is not sensitive.

The third line, the array-line is only needed for two types of gadgets:
Cycle- and Radio-Buttons. Otherwise you write nothing for it.
Be sure to set the correct value for arrays in HUNK:DIMENSIONS!

If you want a Cycle- (Radio-) gadget, the third line could look like
this:
4,Color 1,Color 2,Color 3,Color 4
for a Cycle-gadget that allows 4 choices. The first value sets the number
of choices, the rest of the line are the choices, devided by ",".

1.57 CHUNK:BLANKERINFO

Madhouse reads this Chunk while reading the directory. It contains
some data which is not needed in the current version, but perhaps
I'll add a function to Madhouse to view these data.

WARNING: Madhouse reads the data in this chunk while reading the
Blankers-directory, NOT (!) while opening the BlankerPrefs-window. If
you change something in this chunk, Madhouse will use the old cpu-usage
and stack-values until you don't use the Path-gadget!

```
CHUNK:BLANKERINFO
name
version
cpu-usage
stack

     name = Your Name.
  version = The version of your blanker. Starts with 1, and mustn't be
            a floating-point number. so 1.3 is wrong, but 3 is correct.
cpu-usage = This value can be 0 or 1. If you select "CPU-sensitive
            Random" in the Advanced-Options window, Madhouse will test
            if a program calculates something before it starts a blanker.
            In this case, it won't start your blanker if cpu-usage is 1.
            In other words: if your blanker uses much CPU-performance
            (check it with a cpu-tool), you should enter 1. Otherwise 0.
    stack = The stack depends on your compiler and on your program. Try
            first 5000. If your blanker crashes or aborts, try more...
            (See the documatation for your compiler).
```

1.58 CHUNK:BEVELBOXES

"Bevelboxes" are the boxes without function, but they look like a gadget.
You can see them in the Stars-BlankerPrefs-window.

```
CHUNK:BEVELBOXES
amount
x,y,w,h
[x,y,w,h]
[x,y,w,h]
[...]

amount = The number of Bevelboxes.
     x = x-position of the box.
     y = y-position.
     w = Width of the box.
     h = Height of the box.

After the Chunk-header follows the number of boxes.
Then every box has it's own line.
```

1.59 CHUNK:TEXTS

```
The Text-Chunk must be used for all texts which don't belong to gadgets.
You can use it like the Bevelboxes-Chunk: The number of texts must be in
the first line,

CHUNK:TEXTS
number
color,x,y,Text
[color,x,y,Text]
[color,x,y,Text]
[...]

number = The number of texts.
 color = The color for this text.
     x = x-position of the text.
     y = y-position of the text.
  Text = The text itself. Don't use quotation marks.
```

1.60 CHUNK:LOCALE

```
CHUNK:LOCALE
language1,language2,...

While localizing (giving several languages to a program) Madhouse, I found
out that I need to localize the BlankerPrefs windows, too. Therefore, the
chunks GADGETS and MUI-WINDWLAYOUT need the gadget texts for every
language, not just for one. Two examles:

CHUNK:GADGETS
CYCLE,112,4,170,12,"Mode,Modus",TEXTLEFT
1,DONE

3,"Bouncing Point|Wusel|Random,Springender Punkt|Wusel|Zufall"
SLIDER,112,19,170,12,"Wusel lenght,Wusellänge",TEXTLEFT
5,SLMIN,1,SLMAX,20,DONE
```

```
CHECKMARK,112,34,26,12,"Sound,Toneffekt",TEXTLEFT
1,DONE


CHUNK:MUI-WINDOWLAYOUT
ColumnGroup(2),
   Label( "_Mode,_Modus" ),
   Cycle( "m", "Bouncing Point|Wusel|Random,Springender Punkt|Wusel|Zufall" ),
   Label( "Wusel _lenght,Wusel_länge" ),
   Slider( "l", 1, 20 ),
   Label("_Sound,To_neffekt"),
   HGroup,
      CheckMark( "s,n" ),
      HVSpace,
   End,
End,
```

(Please don’t be frustrated, if this is the first MUI-WINDOWLAYOUT-Chunk you’ve seen in your life: You do not need to understand it if you did not look at the MUI chapters up to now.)

The Locale-chunk concerns all textdefinitions (recognizable with the "-char) in the gadget file. As you can see, every text has a ,-char now. The comma seperates the different languages. The left parts of the texts are the english texts (1st language), the right parts the german ones (2nd language). Other languages could follow. So the right CHUNK:LOCALE looks like this:

```
CHUNK:LOCALE
english,deutsch
```

Madhouse works like this:

- If the MadhouseConfigEd recognizes on startup, that this is no OS 2.1 or higher Amiga, it uses english as the default language. Otherwise it uses the language set with the Locale-editor (from the Workbench), for example deutsch.
- Before opening a BlankerPrefs window, MadhouseConfigEd searches for the CHUNK:LOCALE. If there is none (this Chunk is optional!), it prints the texts into the BlankerPrefs window without any change.
  But if there is one, MadhouseConfigEd looks into the line with the languages and searches the user-defined language ("english" if this is an OS 2.0 Amiga). If it can find this language, it memorizes the amount of commas skipped, and will skip this amount of commas for every text definition. If it cannot find the language, it uses the first one (SHOULD BE "english").

As you can see above, in cyclegadget defintions the comma has a higher priority than the |-separator. Madhouse does the locale-parsing first, and then separates the cycle-entries. For MUI-programmers: the control-keys are texts, too. So you can define different keyboard "maps" for every language, perhaps "_Sound,To_neffekt" and "s,n". If a translation produces the same word ("_Level,_Level" and "l,l"), you don’t have to use the locale-feature and can write "_Level" and "l". Madhouse does no locale-parsing if it cannot find a comma in the string.

Use non-capital letters for the language names, and use the language itself

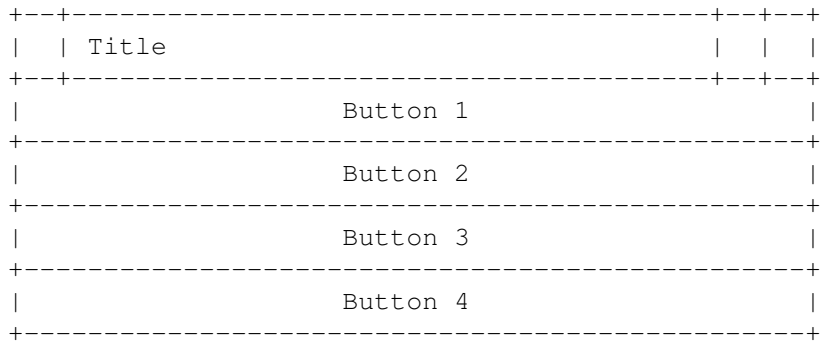
to write the name of it. Examples: français, english, deutsch, ...

I hope I made all things clear - ..

1.61 MUI for MUI-greenhorns.

Creating a MUI-windowlayout is very different. So stop thinking about pixels and fonts, and start thinking about the basic layout of windows.

Here you see a window:



What do you see there? How are the buttons placed? - Vertically. I hope this is the right word, my #?*%-dictionary does not contain the word "untereinander". (If you have a better dictionary, you can look it up :-) But MUI-like we say: The buttons are grouped vertically.

If the window from above was made by MUI, the programmer had just to tell MUI to build a vertical group and give it these four gadgets, and MUI calculated the positions and sizes of the gadgets - very easy.

Perhaps you already know what comes next: horizontal groups containing some gadgets. So we have vertical and horizontal groups and gadgets (of course not just buttons but different types of gadgets, that makes no difference at the moment). But a real window layout is much more complicated. A now comes the point: Groups can contain more than gadgets, they can also contain new groups! It's the same thing with files and directories: a directory can contain files and directories containing even more files and directorys.

< Think a bit. >

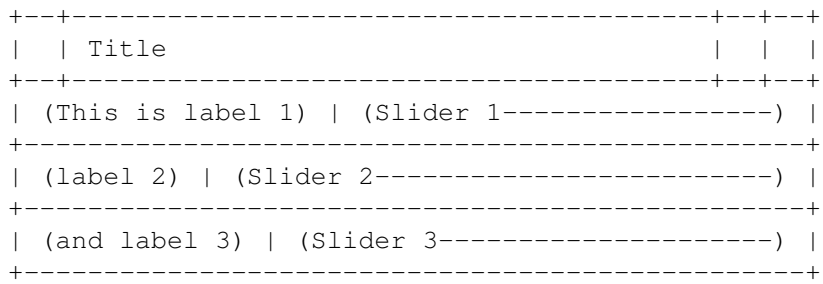
So how looks our window if we replace a horizontal group with two gadgets for button 3? If you think you know the answer, let's see that you are right:

But for real windows we need more gadget types. And these other types have a box containing the gadget itself, like our button, but they also have a text in front of them, like the "Exchange Blanker"-Checkmark and all the others have one. We call these texts "labels".

Usually, you define two objects for every gadget: the label and the gadget itself. Madhouse has some short forms of gadget-definitions con-

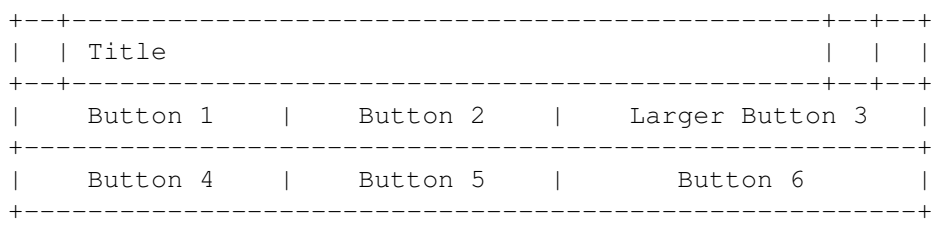
taining also an label on the left side if the gadget, but you use these forms very seldom.

The reason for that is that all MUI-objects (gadgets, labels [and groups]) have some stretching-limitations. You can make a slider as long as you want, but it has a fixed height. This is also true for buttons, string-gadgets and cycle-gadgets. Some objects cannot be stretched at all: labels (containing their text with a fixed width ein height) and checkmarks. Listviews can be stretched in all directions. So if you use the short form of a slider for three sliders in a vertical group, to build three sliders over another with descrptions an their left sides; you will get these sliders, but the left edges of the sliders won't match at all. The window will look like that:



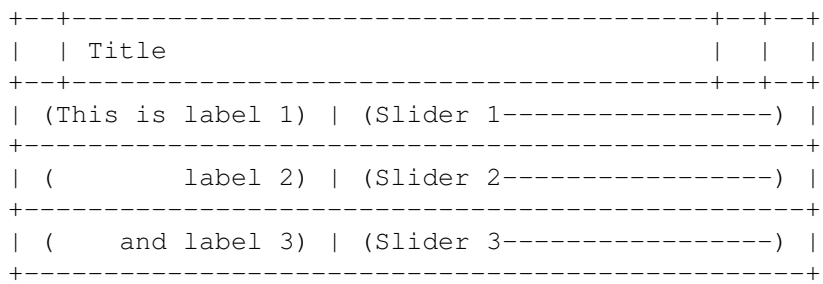
(Note that I marked the dimensions of the objects like that: (-----).)

This does not look very nice. This happens because the labels have a fixed width and the sliders are used to fill the box. In this case, you need column groups. Unlike the other groups, column groups have an argument: the amount of columns. So if you create a column group with three columns, containing 6 Buttons, you will get something like that:



Even if the texts inside the buttons have a different lenght, MUI will group them in a way that every part of one column has the same width.

But we need a solution for our problem above, with the sliders. To have the same width for all sliders, we make a column group with two columns, containing a label, a slider, a label, a slider, a label and a slider (in that order). Then we get



That’s much better! But how does this look like in the gadget-file?
First, I will tell you how our first example looks like:

```
CHUNK:MUI-WINDOWLAYOUT
VGroup,
   Button1,
   Button2,
   Button3,
   Button4,
End,
```

In our new chunk "MUI-WINDOWLAYOUT" we put a vertical group. This group reaches up to "End". It’s the same with IF and ENDIF in a programming language: every ENDIF matches to one IF. Inside the group, we write the objects which are assoziated with it, in this case four buttons. Every line ends with a "," - but you are not allowed to merge two lines to one! You don’t have to "indent" the lines like I did it, but you should do this because it is easier to read. You can use spaces or TAB’s for that.

And how about our column group from above? For didactical purpose, I don’t use a cycle-gadget instead of the third slider.

```
CHUNK:MUI-WINDOWLAYOUT
ColumnGroup( 2 ),
   Label( "_Label 1" ),
   Slider( "l", 1, 10 ),
   Label( "L_abel 2" ),
   Slider( "a", -5, 20 ),
   Label( "La_bel 3" ),
   Cycle( "b", "First entry|Second entry|Third entry" ),
End,
```

Now the chunk contains a column group, as we need one for the correct formatting of the labels. The "(2)" says the column group to make two columns. A column group has to be filled up to the right-most column. In this case, the group can have 2, 4, 6, 8, ... "children". Child is the MUI-word for an assoziated object. If a column group as five columns, it can respectively have 5, 10, 15 and so on children. If you don’t want to fill every place in column group, there is "dummy" object for that, which I will explain later.

As the last example is one which could be written into a gadget-file without changes, you can see there the definitions for labels, sliders and cycle-gadgets. A label is done by the keyword "Label", including the text of the label in this form: Label("Text"),. You can use an underscore _ to underline the following char. This is used to show the user which key of the keyboard can be pressed to control the gadget.
A slider comes next: this type of object needs three arguments: the control-key (which should be marked in the label before), the minimum and the maximum value. So the first example produces a slider which ranges from 1 to 10. You can also use negative values, see the second slider.
The cycle-object needs two parameters: the control key, which is always the first parameter of every gadget, and the different cycle-entries, separated with "|". On my german keyboard, I can find this char next to the backspace-key.

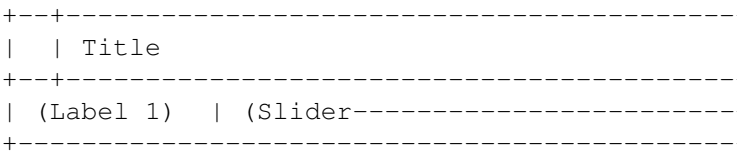
Please note that you cannot use every control-key you want: you can only use non-capital letters and you cannot use the chars o, t and c, because they are already used by the three buttons on the bottom of every Blanker-Prefs-window.

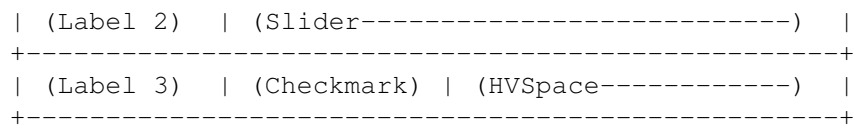
And now comes an example where we need to put one group into another. Please wait patiently while I'm explaining the situation where we need one... (just building up the problem for my solution :-/)

Perhaps (oh, I'm sure...) you have noticed that MUI arranges the gadgets automatically (that is why we don't need pixel-coordinates) and dynamically. This means that almost every MUI window has a size-gadget, which allows us to size it, and MUI recalculates the gadget-positions so that everything fits into the window again. But have you noticed that you can size a lot of MUI windows only horizontally, the height is fixed? (Try some BlankerPrefs-windows). Can you imagine why this happens? Every MUI-object (gadgets and groups, but we'll start with gadgets) has its minimums and maximums for width and height. Sliders, string-gadgets cycle-gadgets and some more have a fixed height (minimum height = maximum height), but no fixed width (maximum width is VERY big). To scale the height of a cycle-gadget would make the cycle-gadget looking silly. Lists have neither a fixed height nor a fixed width, checkmarks have both. So we have answered an half of our question: you can size a lot of MUI windows only horizontally, because a lot of windows contain objects with a fixed height.

But we still need the problem? Here it comes. Try to replace the cycle-gadget by a checkmark (see our latest example, slider - slider - cycle). Now, the result looks quite strange: as the two sliders and the check-mark are placed in the same column, MUI wants to give them the same width. As MUI failes in scaling the checkmark horizontally, it gives the sliders the width of the checkmark, which is of course too small for a slider. The window has no size-gadget at all, because the group inside the window cannot be sized (the left column is blocked by the labels, the right by the checkmark, both objects with fixed width). We have to help MUI, and make the checkmark bigger. (This is ... impossible.) But we could replace the checkmark by a horizontal group, containing a checkmark and a slider. Now, the window could be sized horizontally again, because the column group can be sized again, because it has a column (the right one) which can be sized, because all the members in this column can be sized, because even the horizontal group (containing a checkmark and a slider) can be sized, because this group has one object (the slider) that can be sized. Uff... "can be sized" means here "can be sized horizontally", MUI does the same thing for vertical sizing.

- Hey, my window looks stupid with all these sliders which I filled in after I've heard your explanation! - No problem, MUI offers of course another solution for our problem. It has an object that can be sized to unlimited dimensions and which is... invisible. We call it HVSpace. Our little ANSI-graphic - AmigaGuide has no drawing tools :-(is here:





And you code this like that:

```
CHUNK:MUI-WINDOWLAYOUT
ColumnGroup(2),
   Label( "_Label 1" ),
   Slider( "s", 1, 10 ),
   Label( "L_abel 2" ),
   Slider( "a", -5, 20 ),
   Label( "La_bel 3" ),
   HGroup,
      CheckMark( "b" ),
      HVSpace,
   End,
End,
```

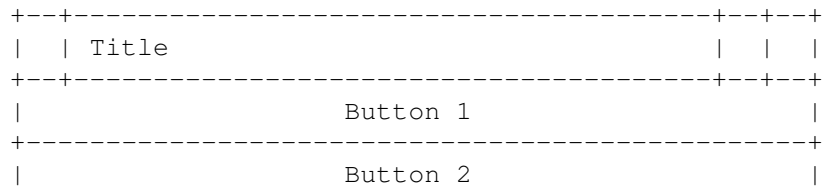
I hope you got it now. The HVSpace is also usefull in a secound case:
if you have a big columngroup and do not want to put objects in aall cells,
you can assign the HVSpace to the cell which should be empty.

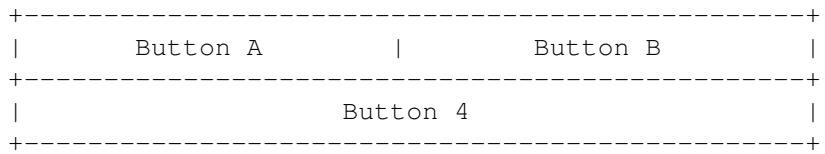
To get further impressions of MUI-layout, you can look into the gadget
files. This was not the whole MUI documentation for Madhouse, just the
basic knowledge. All types of gadgets and another feature of the groups
can be found in the next chapters.

If you have understood the basics of MUI, you can also write your own MUI
application with your knowledge. The code differs a bit (since you are
programming in a real language, and don't write instructions into a small
file for Madhouse). Then, a big compiler goes through your code and not my
self-made MadhouseConfigEd. Where is the difference?! A compiler is a
program which a lot of people work on, for years. My interpreter for the
MadhouseConfigEd was done by myself in four days. Please excuse me,

a) I have not included every MUI-feature into the interpreter. You cannot
   use paged groups, the weight of every object is 100, you cannot use
   CustomClasses (why should you). But you can do a lot. And the MUI
   BlankerPrefs windows generated out of the gadget files look as good as
   'real' MUI applications.
b) I was too lazy to handle ANY error in the gadget-file. You can make
   thousands of errors, and Madhouse does not crash (I hope), but you do
   not get an indiviadual error (or you get no error) for every mistake.
   You will recognize if something is wrong (the window will look strange),
   an it should not be difficult to find the error in such a small file.

1.62 The solution





1.63 Groups

Of course I don't explain the basics of MUI-groups again (see greenhorn chapter).

HGroup and HGroup("Grouptitle")

The HGroup (which places the objects inside horizontally) can have a frame around it and gets visible. The grouptitle will be shown in or above the frame. If you want to have a frame, use the syntax above.

You can do the same thing with VGroups.

ColumnGroup(x) and ColumnGroup("Grouptitle", x)
x stands for the amount of columns, if you use the second syntax you can have a grouptitle.

1.64 The Gadgets

Some standards in this chapter

- Key contains a non-captial letter which is used as the control key for the gadget. The user can control the gadget by pressing this key on the keyboard. Example: Slider("a", 5, 10) makes a slider which can be controlled with "a". You cannot use the chars "o", "c" and "t", because they are already used by the gadgets on the bottom of every BlankerPrefs window.
- Text contains a text which will be displayed on the left side of the gadget. Example: LabelCycle("Color _selection", "s", "Red|Green|Blue") would draw "Color selection" next to the gadget, with the char "s" underlined. This is used to show the user the control key of the gadget.

Slider(Key, Minimum, Maximum)

creates a slider. It ranges from Minimum to Maximum.

LabelSlider(Text, Key, Minimum, Maximum)

like above, but has an additional label (text) on the left side.

CheckMark(Key)

creates a Checkmark.

Cycle(Key, Entries)

creates a cycle gadget with some entries. "Entries" has to contain all entries of the cycle gadget, separated by "|". Example: "RGB|HSV|CMYK". MUI gives the first entry the number 0, so if you select "RGB", Madhouse will write "0" into the prefs file (the same thing with normal gadgets).

LabelCycle(Text, Key, Entries)

like above, but has an additional label (text) on the left side.

String(Key, Length)

creates a string gadget which can contain max. Lenght chars. The upper limit is 500 chars.

LabelString(Text, Key, Length)

like above, but has an additional label (text) on the left side.

Label(Text)

This object gives you the posibility to place your objects better. If you separate gadget and label (as you do it normally), the windows will look much better. Use this object and a gadget without Label... in front of its name.

LLabel(Text)

like above, but this object aligns the text left, not right. This is useful to place another text on the right side of a slider (e.g. "seconds", "minutes", "m", "Objects", things like that). An example of this object can be found in the gadget file of Waves.

HVSpace

The important dummy object, its usage was explained in the greenhorn chapter.

HBar

An elegant object which draws a horizontal bar to separate some gadgets. Should be used only in horizontal groups.

VBar

like above, but draws a vertical bar. An example can be found in the gadget file of Stars.

1.65 Der CHUNK:MUI-PREFSORDER

"How can MUI get along with only one chunk?!" - It can't! In some rare cases, you will need the Chunk MUI-PREFSORDER.

But first a problem to explain its usage:
Madhouse writes the settings of the BlankerPrefs window in that order, in which the gadgets were defined. That is true for normal gadgets and for MUI gadgets. BUT: You can define normal gadgets in the order you want, you cannot do this with MUI gadgets. You have to define them in an order that makes the window look good.
With this Chunk, you can easily re-order the settings which are written into the prefs file. First, you give every gadget a name:

```
CHUNK:MUI-WINDOWLAYOUT
VGroup,
   LLabel( "An extract of Stars" ),
   ColumnGroup( 2 ),
      Label( "_Maximum" ),
      Maximum = Slider( "m", 1, 8 ),
      Label( "M_inimum" ),
      Minimum = Slider( "i", -7, 1 ),
      Label( "C_anges" ),
      Changes = Slider( "a", 0, 15 ),
      Label( "_Start" ),
      Start = Slider( "s", -7, 8 ),
   End,
End,
```

Without MUI-PREFSORDER and the names, your prefs file would look like that:

```
Setting of Maximum
Setting of Minimum
Setting of Changes
Setting of Start
Duration in minutes
Path to the directory of your blanker
```

And now comes the point: if you use the names and write this into your gadget file

```
CHUNK:MUI-PREFSORDER
Changes, Start, Minimum, Maximum
```

you get this prefs file:

```
Setting of Changes
Setting of Start
Setting of Minimum
Setting of Maximum
Duration in minutes
```

Path to the directory of your blanker

Ok, you can live without this Chunk, but sometimes it is helpful.

1.66 Known problems

Problems with AMOS and Protracker (and maybe other Sound-editors)

Some programmers think, there won't be problems if they don't open normal intuition-screens but something else. So did the programmers of AMOS and Protracker. These special screens have a disadvantage:
It's not possible for Madhouse and some blankers to open a normal screen in front of them. So you can only use Madhouse with these programs, if you
- select only Blankers which Aicke made in AMOS (see this doc)
- don't use the password-screen and the Black Background.

Problems with Mousometer

This is an AMOS-problem, too. The AMOS blankers will make superb high-scores, the handler of Mousometer gets mad.

Problems with VGA-monitors

For the same reason (screens etc.) you cannot use programs which were written in AMOS with VGA or Super-VGA monitors. If you use such a monitor, you'll have to de-select the AMOS-Blankers, and Madhouse has got only five blankmodes. Sorry. When the AGA-Update (for which normal Intui-Screens were promised) is available we will change the blankers so you can use them, too. But we suspect that this update will need some time...

Problems with a static Ram-Disk which is mounted as RAM:

Perhaps some people haven't got a normal Ram-Disk which is erased after a reset, but instead a static Ram-Disk (like SD0: or VD0:) which name is RAM:. Then, after a reset, Madhouse will find the RAM:Madhouse_Storage on its startup and it will think it was started twice.
To avoid such a situation, you can easily merge such a line to your S:UserStartup:

```
delete >NIL: RAM:Madhouse_Storage ALL
```

Please notice that it is impossible to have no RAM:-Device at all.

1.67 Authors, Copyright, etc.

We needed nineteen months to develop Madhouse and the Blankmodes. Aicke started earlier, because we planned to create a modular ScreenBlanker while my computer was in repair.

Aicke wrote 62,5% of the blankers, and gave me a lot of tips to improve Madhouse and my blankers.

Madhouse is my second C-program. So there were many bugs in it. I hope all were saved. The first program was the FlyingToasters-Blanker, which I wanted to distribute alone. This english documentation and the german one was done by me. I expect that there are hundreds of faults in this text. It would be nice if someone would correct it...

In the future, we will possibly add some new blankers (I hope our users will add some!). I planned a MUI version of the main program, but as you can see, I done it in the first release...

If you recognize a bug, please tell us. If you wrote a good blanker, don't keep it on your own!! You can distribute it! We would be very happy if we see that even other programmers use our system.

You can send us ideas, graphics and code. And a letter would be very nice, too. If you want a reponse, please insert ENOUGH stemps. (German ones or those which can be used everywhere.)

When sending in bug reports, please state exactly under what circumstances the bug occurred, what equipment was used and what happened. If possible also try to give us enough information to reproduce the bug. It is very difficult to find bugs when you don't know exactly what happened.

Sorry, no E-mail...

Aicke Schulz
Neudeckerweg 118
D-12355 Berlin
(Germany)

If you speak german, you can use Aicke's telephone-number:
030 (Berlin) / 664 48 22

After the 1st of april '96, I can be reached under the following address:

Carsten Jahn
Kuckucksruf 34
16761 Stolpe-Süd
(Germany)

Thank you for your letter!

Registration

You cannot use Madhouse permanently without registering yourself. On start-up, Madhouse won't recognize your preferences files in ENV: and ENVARC:, if it cannot get a correct Keyfile with the filename S:Madhouse.key which you will get after you have sent us DM 20 or US$ 15.

To become registrated, please
- fill in the registration form (you can print it out),
- put it together with DM 20 or US$ 15 in an envelope
  and send it to one of the above addresses (notice that you can write to
  me after april '96 only!)

Let's print the registration form.

Credits: The blankers CrazyPixel, DigitalClock, Drops, Glitter, Memory, Note, Skyline, SoftwareFailure, Thunder and Worldtime were developed with AMOS-Professional. The blankers Fireworks, FlyingToasters, Shuffle, Stars, Soccer and Waves were developed with Maxon C++.

Copyright and Disclaimer:

1.68 Madhouse uses MUI, too!

This application uses

MUI - MagicUserInterface

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY