PicBoot

COLLABORATORS			
	TITLE : PicBoot		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

PicBoot

1.1 PicBoot.guide

```
                    This file describes PicBoot, version 2.8, a program that shows ↩
                       an
IFF ILBM or GIF picture during boot.



                  Introduction
                              What is PicBoot?


                  System requirements
                       What you need to run the program.

                  Legal information
                         Legal information and disclaimer.


                  Usage
                                      Basic usage.

                  Known problems
                            Problems and bugs.


                  StopPicBoot
                               What is StopPicBoot?

                  GetModeID
                                 What is GetModeID?

                  UnpackILBM
                                What is UnpackILBM?


                  Author contact
                            How to reach me.

                  Version history
```

Version history.

1.2 PicBoot.guide/Introduction

Introduction
****

Have you removed all output in your 2.0+ startup, and only see a black screen during boot? Wouldn't it be nice to have a picture instead? A picture that disappeared when the Workbench screen opened?

If so, PicBoot is certainly a program for you. What it will do is to read any IFF ILBM or GIF file containing a picture, and show it. As soon as the Workbench screen appears (or you press any mouse-button), the picture will go away.

Features:

* Fast picture unpacking, using highly optimized assembler. The entire picture is read into memory and then unpacked. This applies both to the IFF and the GIF unpacker.

* Optional auto-detaching; the picture is loaded as fast as possible, with minimal memory fragmentation.

* A picture can be shown a user-specified time after the Workbench screen opens (see
DELAY
and
PATCH
).

* Extremely flexible argument parser.

* Random select among any number of pictures, in several different ways.

* Force a certain display mode for a picture, even if saved with another (can be selected on a picture by picture basis).

* IFF ILBM pictures may be packed with Xpk.

* The comment field of a file may be used to specify options.

* Optional screen centering (horisontally) and border blanking.

* Optional screen fading (in various forms). Uses the increased palette range in AA (24 bits).

* Should work with most "Intuition emulators" for gfx-cards.

* Builtin FixJump, by courtesy of Stefan Sommerfeld.

*
BORDERBLANK
option.

1.3 PicBoot.guide/System requirements

System requirements
****

Apart from OS 2.04+, PicBoot doesn't require any special libraries. The only non-ROM library required is iffparse.library (which normally resides in Libs:).

PicBoot have full support for OS 3.0+ and AA graphics. It should even work with gfx-cards that have an "Intuition emulator".

1.4 PicBoot.guide/Legal information

Legal information
****

This program is freeware. You may copy and use this program freely, as long as the following conditions are met:

* All files are copied in an unmodified state. If additional information is needed, place it in a separate file. Preferably redistribute in the original archive form (*.Lha).

  Exception: So called BBS ads may not be added!

* The copying is done on a non-commercial and non-profit basis only. A copy fee to cover media costs, postage etc. may be charged. This fee may not exceed the fee to obtain an AmigaLibDisk from Fred Fish.

* The copier/spreader is not claiming the Copyright © of this program.

Any exceptions from these restrictions requires written permission from the author, Magnus Holmgren (see
Author contact
).

Disclaimer
==========

Magnus Holmgren neither assume nor accept any responsibility for the use or misuse of these programs. He also will not be held liable for damages or any compensation due to loss of profit or any other damages

arising out of the use, or inability to use these programs.

Magnus Holmgren will not be liable for any damage arising from the failure of these programs to perform as described, or any destruction of other programs or data residing on a system attempting to run the programs. While he know of no damaging errors, the user of these programs uses it at his or her own risk.

1.5 PicBoot.guide/Usage

Usage
*****

To activate PicBoot, add a line to your S:Startup-Sequence, looking something like this:

```
PicBoot Pics:Hi-res/Calvin01.Pic DETACH
```

Or, if you have a list of files in "Work:Text/PicList":

```
PicBoot Work:Text/PicList LIST DETACH
```

Or, if your drawer Pics:BootPics contains some pictures:

```
PicBoot Pics:BootPics/#?.(Pic|Gif) DETACH
```

This line should be located near the beginning in the S:Startup-Sequence (no point in placing it near the LoadWB command, is it? :), but keep it after SetPatch. PicBoot will only output any text if it fails, so don't re-direct its output. Note however that if you place PicBoot before any additional monitors are installed, you'll be stuck with the default.monitor for showing your picture. The DEFAULT switch may be of help here.

Make sure no program makes any output in the CLI window, since then the Workbench screen will open with a boring CLI-window instead...

Options:

FILES
Picture(s) to view, or name(s) of listfile(s).

MODEID
Use this display mode.

PASSWORD
Password for Xpk-encrypted pictures

LIST
The pictures to view are stored in a listfile.

```
CENTER
         Should the screen be centered?

DEFAULT
        Force the default monitor to be used?

AUTOSCROLL
     Enable autoscrolling?

VIDEOOVERSCAN
  Use video overscan?

RTG
            Make PicBoot work with gfx-cards.

WRITEPIXELLINE
 Might make the GIF reader faster.

FADEIN
         Fade speed when opening picture.

FADEOUT
        Fade speed when closing picture.

FADEWB
         Fade speed of Workbench screen after closing picture.

DELAY
          Delay close after Workbench screen open.

BORDERBLANK
    Make picture borders black.

FIXJUMP
        Try to set correct overscan settings before opening  ↩
           picture.

PATCH
          Prevent Workbench from open in front of PicBoot.

DETACH
         Detach from the Shell when picture is loaded.

ACTIVATEWB
     Try to activate Workbench window after close.
```

1.6 PicBoot.guide/FILES

```
                FILES
=====

   This is the only required argument. Here you specify the name of the
picture you want to view. You may enter several files here, in which
```

case PicBoot will select one of them randomly, and show that one.

The name(s) can also be the name of an ASCII file containing a filename list if you specified the
LIST
option.

The name(s) can also be the name of a drawer, in which case PicBoot will randomly select one of the files in this drawer. To use a pattern during this scanning, simply enter the pattern like it had been the name of a file in the drawer. Example:

```
Work:Pics/#?.gif
```

which would make PicBoot select a file ending in .gif in the drawer Work:Pics

If the comment field of the selected listfile or picture starts with "*PicBoot*: " (case sensitive), then the rest of the comment is taken to be arguments, like those in a
LIST
file.

You may freely mix picture and drawer names. Listfiles can only be mixed with the other two ones if the comment contains the
LIST
switch.
In that case, the
LIST
argument should not be used on the command line (or in a list file). Ofcourse you can random select among list files with the "drawer scanner" if you like.. :)

1.7 PicBoot.guide/MODEID

MODEID
======

Short form: M

NOTE: This argument is mainly for the more "advanced" user.

This argument should be a decimal number specifying which screen mode to use. It basically replaces the so called CAMG hunk in an ILBM file (since it contains which screen mode to use). Thus, you must select mode with care, or else the picture will look like trash (nothing more serious can happen. I hope! :). When showing GIF files, it will override the internal "best mode" routines (which aren't good at all. But I haven't bothered to add code to make them better.. :).

To make it easier for you to find out which display mode id to use, there is a small program called GetModeID included, which uses the ReqTools or Asl screenmode requester. Simply select the display mode

you want, and it will print out the number you should use here. See

GetModeID
.

The mode id will be passed through the same validity checking as a normal so called CAMG chunk, so PicBoot should handle bad values properly (although I haven't tested this much.. :).

1.8 PicBoot.guide/PASSWORD

PASSWORD
========

Short form: PW

Here you can specify the password for an Xpk-encrypted IFF ILBM picture.

1.9 PicBoot.guide/LIST

LIST
====

Short form: L

If this switch is specified, PicBoot will interpret the files in the

FILES
argument as names of files containing a list of pictures (or rather, argument lines). PicBoot will then randomly select one of the lines in the selected file, and process it almost like a normal argument line. The only difference is that you can't use the
DETACH
,
DELAY
,

PATCH
or
ACTIVATEWB
arguments. These arguments may not be specified in a listfile (no point in doing it anyway).

The listfile is an ASCII (text) file with a simple layout. On the first line you specify the number of argument lines in the file. This is usually <number of lines in file>-2 (one line is occupied by the count, and the other is the last linefeed). If this value is zero, then PicBoot will exit silently. The rest of the file is simply the argument lines to

choose from. An example:

```
4
Work:Pics/Comics/Calvin02.Pic MODEID 137220
Work:Pics/Comics/Calvin03.Gif
Work:Text/MoreCalvins.txt LIST CENTER ON
Work:Pics/Misc/#?.Gif
```

Note that any arguments specified from the CLI, or in any previous listfile, will be taken as the new default value. In the listfile you may alter this default. This does not include the LIST argument (ofcourse). It is always turned off before parsing a line.

Warning: Since you may enter a new listfile within a listfile, you can be caught in an endless loop, constantly changing (maybe to the same) listfile. No checking for this is done. Also, since there is no CLI-window around, you have no chance to send PicBoot any CTRL-C, if PicBoot should happen to listen to this. You have been warned! :)

Note: A line in a listfile may not be more than 512 chars, or it will be truncated when read. This shouldn't cause any problems I think..

1.10 PicBoot.guide/CENTER

```
CENTER
======
```

Short form: C

Possible arguments: YES, ON, NO, OFF. Default is NO.

If this switch is on (argument is YES or ON), PicBoot will center the picture. This centering should work fine for most screen modes, but one can never know.. :) If a screen promotor is active, then PicBoot can get it wrong (if the screen is opened in another mode than PicBoot had asked for).

1.11 PicBoot.guide/DEFAULT

```
                DEFAULT
=======
```

Short form: DEF

Possible arguments: YES, ON, NO, OFF. Default is NO.

If this switch is on (argument is YES or ON), PicBoot will force the picture to use the default.monitor, regardless of what was actually stored in the picture (in the CAMG chunk). This is needed since very

```
early in the startup, default.monitor is the only monitor available
(e.g.  multiscan.monitor is normally not available). In the future, I
might add more types of "forcing" (e.g. force a picture to PAL, NTSC or
whatever that might be useful).

   This switch also acts on the
               MODEID
                parameter, if specified.
```

1.12 PicBoot.guide/AUTOSCROLL

```
                AUTOSCROLL
==========

   Short form: AS

   Possible arguments: YES, ON, NO, OFF. Default is NO.

   If this switch is on (argument is YES or ON), the OS 2.0+
autoscrolling of screens will be enabled.

   Note: During boot, this switch may make the actual display a bit
smaller than normally possible. There is nothing I can do about that...
:) You can, however, by ensuring that ENV:/IPREFS is properly set up
before PicBoot is started. Or you could try the
                VIDEOOVERSCAN
                 switch.
```

1.13 PicBoot.guide/VIDEOOVERSCAN

```
                VIDEOOVERSCAN
=============

   Short form: VO

   Possible arguments: YES, ON, NO, OFF. Default is NO.

   If this switch is on (argument is YES or ON), the visible size of the
opened screen will be as large as the system can handle (assuming the
picture is large enough). Forces
                AUTOSCROLL
                 to YES.
```

1.14 PicBoot.guide/RTG

```
                RTG
===
```

   Possible arguments: YES, ON, NO, OFF. Default is NO.

   If this switch is on (argument in YES or ON), then PicBoot will do things a little differently, in an attempt to make it work with gfx-cards (it have been tested with Picasso II). The main difference is that the screen is opened first, and the picture is decoded into this screen (usually the picture is decoded first, and then the screen is opened). Thus, you should only use this switch if the picture should be shown with the gfx-card rather than the native Amiga graphics. If the gfx-card isn't used, the picture decoding may be a little slower.

   Oh, btw, RTG stands for ReTargetable Graphics.

   Note: This feature requires OS 3.0 to work. If you don't have OS 3.0, then this argument is ignored.

   Note: You might need to specify a new
```
                MODEID
```
 in order for PicBoot to use gfx-card screen.

1.15 PicBoot.guide/WRITEPIXELLINE

```
                WRITEPIXELLINE
==============
```

   Short form: WPL

   Possible arguments: YES, ON, NO, OFF. Default is NO.

   If this switch is on (argument is YES or ON), then PicBoot will use a ROM function to convert/write the pixel data of a GIF picture. Please note that this only have any effect if
```
                RTG
```
 have been used, and the opened screen isn't a native Amiga screen.

   In some cases, this can make PicBoot faster, but in others, it can make PicBoot slower. There is an explanation for this, but it is a bit technical. If you don't understand it (or you don't know enough about your gfx-card), then I suggest you test it a little, finding out which is fastest for different pictures.

   Most gfx-cards have a "chunky" screen. That is, each pixel is stored in one byte, which specifies which color (in a palette) that the pixel have. This is different from the native Amiga screen that have bitplanes (where the color number is spread over several bytes). Since a GIF picture stores the data in a chunky format, it would be a waste of time to first convert the chunky data to bitplane form, and then

back again, if it should be displayed in a chunky screen.

In the case of a chunky screen, this switch can improve the speed quite a lot, if a certain ROM function is patched by the Intuition emulator. This patch should simply write the chunky data directly to the right area of the screen.

However, not all gfx-card screens are chunky (at least the Picasso II stores 2-16 color screens as bitplanes), and perhaps the "Intuition emulator" haven't patched the above mentioned function. In those cases, then this switch will make things slower (since the original function (which converts the chunky data to bitplane form) is quite a bit slower than the routines in PicBoot).

To test which is fastest, use something similar to the following commads:

```
PicBoot Pics:Gifs/Island.gif RTG ON DETACH
PicBoot Pics:Gifs/Island.gif RTG ON DETACH WPL
```

and measure how long time both commands took to run (note that PicBoot detaches after the picture have been completely decoded), and only use the WPL switch if it made PicBoot faster.

1.16 PicBoot.guide/FADEIN

```
FADEIN
======
```

Short form: FI

Argument range: 1 to 4. Default is no value.

This value specifies the speed with which the picture should fade in when the screen is opened. HAM pictures can't be faded.

1.17 PicBoot.guide/FADEOUT

```
                FADEOUT
=======
```

Short form: FO

Argument range: 1 to 4. Default is no value.

This value specifies the speed with which the picture should fade out when the screen is closed. HAM pictures can't be faded. Only useful in combination with

```
                DELAY
```

(otherwise the picture will be in the back,
were the fade isn't visible! :).

1.18 PicBoot.guide/FADEWB

FADEWB
======

Short form: FWB

Argument range: 1 to 4. Default is no value.

This value specifies the speed with which the Workbench screen
should fade in when the picture screen have been closed. Intended to be
used in combination with the
FADEOUT
(and
DELAY
) argument(s).

Note: If you have a Workbench screen with lots of colors, this
fading can cause problems, if you have any background pattern. When the
system loads these patterns, it might try to allocate some of these
colors for use by the pattern. If this happens while PicBoot is
actually doing the fade, it is easy to imagine that the result won't
quite be what you expect (i.e. wrong colors in the pattern). If this
happens for you, simply don't use this feature.

1.19 PicBoot.guide/DELAY

DELAY
=====

Short form: DL

A "problem" with PicBoot is that the Workbench screen first opens,
and then processes the Sys:WBStartup drawer, which takes a little time.
This means that the picture PicBoot shows disappear before the boot is
complete. To avoid this problem, the DELAY switch can be use to specify
the number of ticks (there are 50 ticks each second) PicBoot will wait
after the Workbench screen have opened.

However, this isn't perfect. When the Workbench screen opens, the
PicBoot screen must be brought back to the front again. This causes a
little "flicker". To avoid this, use the
PATCH
parameter as well (this
feature requires OS 3.0+ to work).

There is a special delay value, 0, which causes PicBoot to wait until you either press any mouse button, or another program sends PicBoot a break signal (CTRL-C). The program

StopPicBoot

was written to do this.

1.20 PicBoot.guide/BORDERBLANK

BORDERBLANK
==========

Short form: BB

If this switch is specified, then the special border blanking mode will be enabled, which can make the picture look a little better. This only works on OS 3.0 or better.

1.21 PicBoot.guide/FIXJUMP

FIXJUMP
=======

Short form: FJ

If this switch is specified, then PicBoot will try to set the correct overscan settings before opening the picture. This should fix the "jumping" that otherwise occures when IPrefs is started.

What it does is simply to read the file EnvArc:Sys/overscan.prefs, and sets the proper values.

Important note: This is a hack, since it calls an undocumented system function. The code seems to work well on my system (A4000/040 OS 3.1), but I don't make any guarantees.

1.22 PicBoot.guide/PATCH

PATCH
=====

Short form: P

If this switch is specified, PicBoot will install a patch in Intuition, so that the Workbench screen (or rather, any screen opened, that explicitly doesn't say that the screen shouldn't open behind the

others) doesn't open in front of the PicBoot screen. This removes the "flicker" that normally occurs when using the
DELAY
option. For this option to be useful, the
DELAY
parameter must be used as well.

Note 1: This option only works on OS 3.0 or higher. This is due to the OS (as far as I know), and there is nothing I can do about it (tech note: Workbench in OS 2.0x doesn't seem to call the open screen function via the external library vector).

Note 2: This kind of patching is not a recommended thing to do. Programs should not do temporary patches like this. However, to avoid the flickering, there is no alternative.. :)

Note 3: In case some other program patches the same function after PicBoot have installed its patch - and you don't have a program like e.g. SetMan installed - then PicBoot will leave a small memory allocation behind (6 bytes), to avoid any problems.

1.23 PicBoot.guide/DETACH

DETACH
======

Short form: D

If this switch is specified, PicBoot will detach from its calling CLI when the picture is fully loaded and displayed. If you specify this option, you shouldn't "Run" PicBoot. This option will reduce memory fragmentation, and will ensure that the picture gets loaded quickly. I don't think this feature will cause any problems, but I added the switch just in case.

1.24 PicBoot.guide/ACTIVATEWB

ACTIVATEWB
==========

Short form: AWB

If this argument is specified, then PicBoot will try to activate a Workbench window after closing the picture. This might be useful if you are using the
DELAY
argument.

1.25 PicBoot.guide/Known problems

```
                Known problems
**************
```

I do not know of any real bugs in PicBoot. However, certain parts of the program may still contain bugs. E.g., pictures that have a mask bitplane (mskHasMask) are supported, but since I only have one (compressed) picture that have a mask, there might be a bug in that code (can't test it properly). Please report any problems!

Currently there is no support for SHAM, PCHG and similar pictures. These pictures aren't that common, and I have an Amiga with AA-graphics, so... :) Color cycling is currently ignored (I have no need for it).

Interlaced GIF pictures aren't supported at the moment. The decompression of such pictures would be slower, and there are tools that can de-interlace a picture.

PicBoot doesn't remap GIF files in any way. Even if you have ECS, GIF files can still be useful. This is because a GIF file can have from 2 to 256 colors (inclusive). Thus, if you have a program that can save a 16-color picture as a 16-color GIF file, there will be no problem to view it with PicBoot.

PicBoot doesn't make use of any "chunky to planar" hardware, if it should happen to be installed (e.g. Aikiko). Anyone who have it, so I can test it if I should decide implement it? :) It would be fairly simple to do, since my own (rather fast, I might add :) chunky to planar routines have very similar restrictions.

The "best mode" routine used in the GIF reader isn't good at all (this includes the ROM function in OS 3.0+! :). I suggest you use the

```
MODEID
```

parameter instead (Correction: The ROM function isn't good when there are several different monitors to choose from. If only one or two (similar) monitors are available, then the result is usually rather good).

The "monitor promotion" (i.e. the

```
DEFAULT
```

parameter) is not very intelligent.

Pictures with more than 8 bitplanes are currently not supported by PicBoot.

The centering for (some?) Super72 screens doesn't work. I suspect this is an OS-"bug" (I know that PicBoot calculates a reasonable offset, which Intuition seems to ignore).

1.26 PicBoot.guide/StopPicBoot

```
                StopPicBoot
***********
```

   StopPicBoot is a small program that simply tells PicBoot to quit, if it should happen to be in memory. This is useful in combination with the

```
                DELAY
```

 option. If this is set to 0, PicBoot expects someone to tell it when it is time to exit, and this is what StopPicBoot does.

   By having StopPicBoot in Sys:WBStartup, then PicBoot will close its screen when the boot process almost complete (the tooltype STARTPRI should be very low (-120 or so), so that StopPicBoot is started as the last program)

1.27 PicBoot.guide/GetModeID

```
                GetModeID
*********
```

   GetModeID is a simle program that shows a Asl or ReqTools screenmode requester, whichever is available. The program will then print out the decimal identifier for the selected screenmode, suitable for use together with the

```
                MODEID
```

 parameter. This program can only be used from a Shell. Example usage:

```
       PicBoot Island.Gif MODEID `GetModeID`
```

   This will first show a screenmode requester (if you have one, that is), and then show the GIF-picture in the selected screenmode.

1.28 PicBoot.guide/UnpackILBM

```
UnpackILBM
**********
```

   UnpackILBM is another simple program (at least in theory... :). It will take any IFF ILBM picture and unpack the so called BODY chunk in it (this is the actual image data). This means that e.g. PicBoot will be able to display that image a little faster, at least if loading it

from some fast media. Or you could repack the picture with
PowerPacker/Xpk, to maximize the compression (as the
compression used in IFF ILBM isn't a very efficient one. But on the
other hand, it is rather fast and simple). Example usage:

```
UnpackILBM Island.Pic Island.Pic.NoComp
UnpackILBM Island.Pic
```

The first example till unpack the picture to a new one, while the
other will - via a temprary file - overwrite the original picture with
the uncompressed version.

Note: I haven't tested this program that much. I've converted a few
pictures, so it seems to work fine (at least when there aren't any
errors), but one can never now.. Please report any problems!

Note: This program doesn't strip any information. All chunks will
remain. The picture data is only decompressed.

1.29 PicBoot.guide/Author contact

```
Author contact
**************
```

PicBoot was written by Magnus Holmgren. If you have any comments
etc, feel free to send me a note. You can reach me via internet on this
address:

```
cmh@augs.se
```

Fido-net messages should go to "Magnus Holmgren",
2:204/404.6@fidonet.org. Snail mail should reach me if you write the
following address on the envelope:

```
Magnus Holmgren
Kvarnbergsvägen 4
S-444 47  Stenungsund
SWEDEN
```

1.30 PicBoot.guide/Version history

```
               Version history
***************
```

Version 1.00-1.03

Version 2.0

Version 2.1

Version 2.2

Version 2.3

Version 2.4

Version 2.5

Version 2.6

Version 2.7

Version 2.8

1.31 PicBoot.guide/Version 1.00-1.03

```
Version 1.00-1.03
=================

   Ancient versions.
```

1.32 PicBoot.guide/Version 2.0

```
                Version 2.0
===========

   Release date: 29 Mar 94

   *  BLACK argument removed. Not needed any more, since PicBoot now
      will first allocate the needed memory, decode the picture into
      this memory, and then open the screen. This makes the screen
      opening/closing a little faster too (practically instaneous on my
      A4000/040).

   *  Pictures (or rather, brushes) that were less than 16 pixels wide
      wouldn't decompress properly... :)

   *  Added support for the CMAPOK flag in the BitMapHeader.bmh_Flags
      (previously called bmh_Pad) field (if this flag is set it
      indicates that the color map contains 8 bits/color rather than 4
      bits/color).

   *  Major code cleanup. Made the program somewhat larger, but... :)

   *  GIF support added. Should be a little faster than PPShow.. :)

   *  The
                MODEID
```

```
                argument wasn't properly "passed on" to any following
     listfile(s).

   *  Rewrote rtGetModeID into GetModeID, that first checks for Asl,
     and then tries with ReqTools before giving up. This new version is
     in C, compiled with DICE, without any startup code, and is fully
     residentable. :) See
                GetModeID
                .

   *  Included UnpackILBM, that takes any IFF ILBM file (with a BODY
     chunk, i.e. a normal picture) and writes it with an uncompressed
     BODY instead. Written upon user request. See
                UnpackILBM
                .
```

1.33 PicBoot.guide/Version 2.1

```
                Version 2.1
===========

   Release date: 14 May 94

   *  The
                DEFAULT
                 parameter didn't do anything. Fixed

   *     UpackILBM and GetModeID updated a little. Version string added,
     recompiled with DICE 3.0 and some other minor changes.
```

1.34 PicBoot.guide/Version 2.2

```
                Version 2.2
===========

   Release date: 12 Jul 94

   *  If the listfile was too short (i.e. not enough number of lines in
     it), PicBoot would crash.

   *  Made the detaching code more system friendly. I hope this will
     fix the problems a few users have had.

   *  A few minor bugs fixed + some minor optimizations...

   *  If the number on the first line in the listfile is 0, then PicBoot
     will exit silently. Now why did I add this... >;)
```

```
   *  Improved the random number algorithm.

   *  Added the
                DELAY
                 parameter.

   *  Added the
                PATCH
                 parameter.

   *  Tweeked the GIF-unpacker a little. Found yet another Macro68
      (V3.170) bug while doing that.. :/ (Watch out for bra.l to other
      sections/modules when generating code for the 68020+. The branch
      target is not correct. :)
```

1.35 PicBoot.guide/Version 2.3

```
Version 2.3
===========

   Release date: 30 Aug 94

   *  Rewrote startup code and argument parser in C, for easier
      maintainance (and to simplify the implementation of some of the
      features below).

   *  You can now also specify a directory (with optional pattern
      matching), and PicBoot will randomly select among the files found.
      As usual, you can use this feature whereever PicBoot used to
      expect a file name.

   *  If the comment field of a file that PicBoot will read (i.e. a list
      file or a picture) starts with the string "*PicBoot*: " (case
      sensitive), then the rest of the comment is assumed to be
      arguments, to be parsed like they had been found in a list file.

   *     UnpackILBM will not delete the temp file if it couldn't be
      renamed to the original.

   *  Removed a piece of debug code in the OpenScreen() patch (it
      flashed the screen). Harmless, but annoying.. :)
```

1.36 PicBoot.guide/Version 2.4

```
                Version 2.4
===========

   Release date: 11 Oct 94
```

```
   *  Added the arguments
              FADEIN
              ,
              FADEOUT
               and
              FADEWB
              , to make various
     color fades when the screen is opened/closed.

   *  Modified the random routines again (changed the seed source). I
     hope it works better now.. :)

   *  If
              DETACH
               was used and a relative filename was used, then the file
     wasn't found (except in some cases).

   *  Using
              PATCH
               in combination with
              DELAY
               on pre-OS 3.0 systems
     caused a "Not enough memory" message to be printed, and the picture
     wasn't showed (the argument should silently be ignored).

   *  Some error messages lacked a final linefeed char (an autodoc was
     a bit misleading.. :).

   *  The
              DELAY
               was incorrectly interpreted as seconds, and not as
     ticks.

   *  Closing the picture before a
              DELAY
               timeout had expired caused a
     crash.
```

1.37 PicBoot.guide/Version 2.5

```
              Version 2.5
===========

   Release date: 13 Oct 94

     The Workbench screen wasn't "unlocked" in some cases (when either

              FADEWB
               or
              ACTIVATEWB
               had been used).
```

1.38 PicBoot.guide/Version 2.6

```
                Version 2.6
===========

   Release date: 30 Nov 94

   *  Added
                RTG
                 switch, which will make PicBoot attempt to be more
     compatible with various "Intuition emulators" for gfx-cards. Don't
     use this switch if you don't need to; it will cause the screen to
     be opened before the rendering (possibly making it slower).

     Also added
                WRITEPIXELLINE
                , which may improve the speed of the GIF
     reader sometimes, when using the
                RTG
                -mode.

     Thanks go to Roger Westerlund, who made me actually try to make
     PicBoot
                RTG
                 friendly, and also did all the needed testing (I don't
     have any gfx-card :).

   *  Changed the screenmode id "validator". Needed for better
                RTG
                     compatibility.

   *  Some fade related code was a bit broken, causing problems with
     e.g.  HAM-pictures.

   *     UnpackILBM is now able to decompress via a temporary file if
     the file and the current directory are on different volumes. Also
     improved some error reporting.

   *     GetModeID now filters out some useless modes (the dual payfield
     (DPF) ones). Also, the ReqTools screen mode requester didn't show
     "non standard" modes (such as HAM and EHB).
```

1.39 PicBoot.guide/Version 2.7

```
                Version 2.7
===========

   Release date: 17 Jul 95
```

```
   Well, I started working on this version a couple of months ago
(making it almost finished), but then I forgot all about it.. :)

   *     PicBoot didn't unlock the drawer after a directory scanning.

   *     PicBoot could hang if a
                 DELAY
                  was requested, and the picture
     was closed before the Workbench window had opened.

   *     PicBoot didn't handle names with spaces in them very well..

   *  Fixed some bugs in the IFF reader that could cause crashes.

   *  Added support for Xpk packed/encrypted IFF files.

   *  Made GetModeID a bit smaller. Didn't think twice before adding the
     mode filter.. :)

   *     UnpackILBM would report an error and delete the file if the
     same file was specified in both FROM and TO. Now only the error is
     reported.
```

1.40 PicBoot.guide/Version 2.8

```
                Version 2.8
===========

   Release date: 02 Nov 95

   *  Added the
                 FIXJUMP
                  switch. Removes the "jumping" that usually
     occures when IPrefs gets loaded. Thanks to Stefan Sommerfeld for
     sending me the sources to his program FixJump (which this
     implementation is based on).  Note that this is a hack, and may
     not work on all systems.

   *  Made a few changes, in an attempt to fix the fading problems a
     couple of users have reported.

   *  Added the
                 BORDERBLANK
                  switch.
```