NewPattern

Daniel Balster

COLLABORATORS			
	TITLE : NewPattern		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Daniel Balster	June 15, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

NewPattern

1.1 main

NewPatterns 1.0

a Workbench Backfill Replacement

written by

Daniel Balster

Ok

1.2 index

This is (currently) FreeWare

1.3 help

currently empty.

1.4 intro

Introduction

You all know the WBPattern application in the SYS:Prefs drawer. You can select a small pattern or an image for the background of the ROOT, DRAWER and SCREEN windows. But the backfill method of the Workbench is slow, and if you use MagicLayers or some other kind of opaque window movers you will know that. Just open a drawer and size it to the maximum size, and then click on its ZOOM gadget and minimize it. Now play with the ZOOM gadget and try it with several Workbench resolutions. Is that fast ?. I think not. And this ugly WBPattern application! I have so many patterns I want to have as a background. And each time select it with that? No. I've tried MagicSelector, and after a while I wrote my own Random-Pattern-Selector (MakePatPrefs, useless now). But it was still the speed of the refresh that makes me feel sick. Then I found the program LayerHook on the aminet and that method was so speedy, that I decided to write a hack that replaces the original backfill hook. And another thing was important to do so: the original hook isn't able to handle real 24 Bit! But my CyberGraphX Workbench runs sometimes in

15/16/24 Bit and so I wanted to have truecolor backgrounds. This isn't implemented yet, but a working version exists; it looks and feels great! Read this documentation (care)fully, and mostly the bug section.

Daniel Balster

1.5 required

Really Required

- Workbench V39 and Kickstart V39 (AmigaOS 3.0) or higher

- 68020++ processor (A1200/A4000/CD32/A3000)

- datatypes.library and a couple of datatypes, of course

- as much ChipRAM/FastRAM as needed by the patterns.

- a mouth (without you cannot eat the food)

Recommended

- AGA

- 4 MB FastRAM and 2 MB ChipRAM

Professional

- a graphics card

- CyberGraphX (if you want to use 15/16/24 Bit Patterns)

- 16 MB FastRAM (used to hold more than one 24-Bit fullscreen image)

- 68040/68060 Processor (JPEG decompression needs time)

1.6 install

Patterns

Ofcourse you will need some patterns and backdrops somewhere on your harddisk. Without it doesn't make sense! Recommended location is SYS:Prefs/Patterns/

Database

If you like you can create a simple database of all your patterns. Just write a TEXT file and use each line as an entry for a pattern, i.e.

SYS:PREFS/Patterns/NewIcons-F/Foilage SYS:PREFS/Patterns/NewIcons-F/FrankLloydWright SYS:PREFS/Patterns/NewIcons-G/G-rain SYS:PREFS/Patterns/NewIcons-G/GearBox SYS:PREFS/Patterns/NewIcons-G/Granite-01

and save it to a location where you can find it, like S:patterns.database. This is used by the program RandomPattern.

Program binaries

There are currently three shell executables.

NewPatterns RandomPattern SetPattern

Copy them to a location in your command path, like C:

Automated usage

If you want NewPatterns to be installed permanently, you must edit your S:startup-sequence. Add these lines RIGHT BEFORE LoadWB:

; used for a constant pattern set NewPatterns SetPattern >NIL: ROOT <pattern> DRAWER <pattern> SCREEN <pattern>

or

; used for a random pattern set NewPatterns RandomPattern >NIL: <pattern.database> [ROOT] [DRAWER] [SCREEN]

See the future link future} and bugs link bugs}

1.7 usage

NewPatterns RandomPattern SetPattern

The Versions that ends up with .cgfx are compiled with the CyberGraphX stuff.

I've included two silly RAW images. Their format is 200x150x24 Bit RAW RGB. This is what the backfiller only will accept. WARNING! DO NOT LOAD *ANY* OTHER PICTURES OR FILES WITH LESS THAN 90000 BYTES! This is only for test purposes and not for a regulary use.

Switch the Workbench into 15/16/24-Bit mode and enter at the shell prompt:

>SetPattern ROOT logo.rgb DRAWER daniel.rgb

logo.rgb is a video grab of a SGI mousepad; I used the built-in desktop camera of an INDY. The mousepad is nice, eh ?

daniel.rgb is not so nice. This shows me getting bottles of beer out of an fridge. This was cropped out of an BASE*4 PhotoCD image.

Note: do not do "Mode Jumping" with the Workbench, I haven't implemented the screennotify.library support!

1.8 newpatterns

NAME NewPatterns

AMIGADOS NewPatterns

FUNCTION Installs the replacement backfill hook for the Workbench. Once started it detaches from the shell.

INPUTS none

RESULT none

NOTES Once started, there is not possibility in removing it. Yet.

1.9 setpattern

NAME SetPattern

AMIGADOS SetPattern ROOT,DRAWER,SCREEN,NOROOT/S,NODRAWER/S,NOSCREEN/S

FUNCTION Changes a pattern for the desired object.

INPUTS ROOT - name of a file to be used as ROOT pattern (Workbench Window)

DRAWER - name of a file to be used as DRAWER pattern (Workbench Drawers)

SCREEN - name of a file to be used as SCREEN pattern (Workbench Screen) THIS IS CURRENTLY DISABLED

NOROOT - set the ROOT pattern to an empty pattern (overrides ROOT)

NODRAWER - set the DRAWER pattern to an empty pattern (overrides DRAWER)

NOSCREEN - set the SCREEN pattern to an empty pattern (overrides SCREEN) THIS IS CURRENTLY DISABLED

RESULT The Workbench should be refreshed with your new patterns

NOTES Even if no patterns are provided, the Workbench will be refreshed.

1.10 randompattern

NAME RandomPattern

AMIGADOS RandomPattern DATABASE,ROOT/S,DRAWER/S,SCREEN/S

FUNCTION Chooses random patterns for each of the given objects.

INPUTS DATABASE - a Textfile containing filenames (full path!) of all your patterns (or a selection)

ROOT - select a pattern for the Workbench Window

DRAWER - select a pattern for the Workbench Drawers

SCREEN - select a pattern for the Workbench Screen THIS IS CURRENTLY DISABLED

RESULT The Workbench should be refreshed with your new patterns

NOTES

1.11 registration

NewPatterns will be Shareware.

Currently it is not crippled in any way, but this will change in the next version. All NewPatterns versions below 2.0 won't be crippled in any way, and won't be supported by forthcoming versions.

This version is just a PREVIEW. I published it to get YOUR reactions. It is much work to develop a complete product and to maintain it. I need to know if there are people who has also a need of this.

If you are interested in it, DROP ME A MAIL .

1.12 author

If you contact me by EMail you should use the topic "NewPatterns - <special>" where <special> is a short set of keywords the mail is about.

If you want to talk/chat with me, try the IRC or just visit me.

SMail Daniel Balster Max-Reger Weg 48 33100 Paderborn

EMail dbalster@uni-paderborn.de

IRC dbalster#amigager

1.13 future

The next version will have these features

- Make it a WBStartup application

It is required that the Workbench screen is initally open. Currently the patches are launched right before the LoadWB command.

- SCREEN Support

The method the SCREEN is backfilled differs from the method the Windows are. Is this really required? I mean the SCREEN is always hidden by ROOT. I do not need this. Do you ?

- 15/16/24 Bit image loaders.

The hook is capable of copying/blitting with WritePixelArray() and BltBitMap() on CyberGraphX screens (use the .cgfx versions to test it), but I haven't implemented a stable set of 15/16/24 Bit loader routines yet.

This is still future:

The internal format of the image will be adapted to the Screen it will be used for. For an example: a 24Bit images will be converted into a special 15 Bit format (assuming you use a 15 Bit Workbench). This will reduce the memory that is needed for the picture cache and speed up rendering a lot. Truecolor remapping (Pixelformat remapping) is much faster then 8Bit dithering.

- A nice preferences Editor with a Standard GUI for all the Options.

This includes locale.library, font sensitive GUI and online help. Producing such a GUI is not difficult but it takes much time...

- Image Cataloger

I will implement a fast Image cataloger in the Preferences Editor, that will show small thumbnails from all the images. If the editor runs on a CyberGraphX Workbench with 15/16/24 Bit

- Transparent Patterns

This is a *BAD* hack; it uses the hidden areas of a window as a background pattern - this produces a somehow "transparent" effect. The parsing through the damage/savelists of the layers.library is obsolete and I don't know if this will be released ever as it has still many bogus sideeffects.

- Timed Patterns

With this feature it will be possible to connect dates with patterns. Just think about it: Each morning a rising sun pattern and a moon pattern during the night! If you have birthday, a cake pattern will appear and on christmas all windows will have snow... how sweeeet.. ;-)

I wrote this as a script months ago, and it is quite nice ;-)

- Parametric Patterns

Got tired of all your pattterns ? This will produce nice looking parametric patterns from editable formulas.

- user requested

Write me what YOU want to have implemented.

1.14 credits

NewPatterns uses the routine CopyTiledBitMap() from the LayerHook.lha archive. The authors of that package can always request a free version of NewPatterns.

This routine was adapted to work with CyberGraphX by me

1.15 config

You can currently control only the picture remapping. This is done by using ENV: variabels.

Create ENV:NewPatterns for this, and also ENVARC:NewPatterns if you want to save the options.

NewPatterns/MAXCOPYWIDTH NewPatterns/MAXCOPYHEIGHT

These variables were used by the CopyTiledBitMap() routine. They were used to determine the optimal size for a blit.

The Standard settings are 256x256 for Width and Height. This is a good size. If you use ECS/AGA and have the pattern in the ChipRAM, BltBitmap works much faster the higher this value is. On custom graphic boards, pattern data must be copied with the CPU. If your board is slow (like the Picasso) you want to set this values to minor ones. Within the board memory the graphics boards' internal blitters are much faster, so you can control this bottleneck with this option.

NewPatterns/REMAP

Set it to 1 if you want to remap your images to the colors of your Workbench. This is the default.

Set it to 0 if the images shouldn't be remapped at all. This is useful for MagicWB owners, who remapped all their patterns to a default palette. This speeds up loading an image.

NewPatterns/PRECISION

Unlike the internal Workbench backfiller, this allows you to set the remap precision. This affects the number of pens to be allocated by a picture and the remapping time. Set it to

3 - exact remapping is the best and pen exhaustive method. Takes time. 2 - image remapping is the default method. 1 - GUI remapping is not as good as the above. 0 - Icon remapping uses the least number of pens and is the fastest remapper.

All other values cause the default precision.

WARNING! This all are experienced user flags. In circumstances it is possible that a pattern couldn't be loaded anymore. In such a case the Backfiller returns to a standard Clear() Hook.

1.16 bugs

I've tried to provide an earthquake proof method in implementing this. Enforcer was used to check all oddities in conjunction with pointers, and several memory snappers helped me to locate memory losses.

However, call me if you detect an error.

List of killed bugs:

- MagicLayer and MCP/MoveFullWindow produced layer garbage. fixed.

- Depth arranging of windows sometimes produced "offset garbage"; the pattern was refreshed with a different offset. fixed.

List of known bugs:

- Ooops. If you are using CyberGraphX, you cannot use the CyberGraphX version with normal AGA/ECS Screens - I check for a valid CyberGfxBase before blitting -> AGA/ECS screens will be blank instead of filled. I won't fix this in this version; it is a good demo restriction ;-)

- Do not use the AGA version on 15/16/24 Bit Screens; it is REALLY slow.

- the CyberGraphX version sometimes hangs while loading.

- The SCREEN pattern support is not completely compiled in this version; it was buggy (it produces garbage). I use InstallLayerInfoHook() on the Screen's Layer_Info, but that doesn't seem to work properly.

- Refreshing the Workbench is fast but not perfect. I notify IPrefs with a dummy wbpattern.prefs file to cause a refresh. I also tried a hack, that sends a modified Intuition Message to the Workbench; but this causes still Enforcer hits and has been disabled for now.

- On a 15/16/24 Bit Workbench the above mentioned method to refreshing the Windows won't work. This requires still some work; the oddity is that it sometimes work. (try "Setpattern" twice (no parameters)

- While in 15/16/24 Bit Mode, loading files not in 200x150x24 raw RGB format *WILL* crash/deadlock your machine. You have been warned.

- If the Workbench window is NOT a backdrop window, it will be handled as a drawer window. This is because I check for WFLG_BACKDROP & WFLG_WBENCHWINDOW in the win->Flags. This is quite fast and system friendly, and can be figured out during OpenWindowTagList(). Is this really a bad bug?

- During a pattern change, the colors of the pattern that is going to be replaced will look ugly. This is becaused I free the pens and use them for the new image. This gives you more free pens! I could clear the pattern during the exchange, of course. But that is too much overhead; with the flickering you get the fastest speed and the best pens. Who cares? It took less than a second!

- This guide was written by a german; it surely contains many many errors and other language faults... But how many in the world can read german guides?

- If you change the screen resolution or close the Workbench, all images lost their pens and will look ugly. I instruct the datatypes.library to remap the images to a new bitmap and then FREE THE SOURCEBITMAP, and therefore it is NOT possible to remap the images to a new screen. They must all be loaded again. And even this job is not implemented yet, since I currently work on it. I will use the screennotify.library for this.

- After closing/opening the Workbench I got three mysterious crashes until now (3 out of 1000 ;-) and they could have been caused due to the non-existing screennotify.library support. I do not know if I must free all pens BEFORE closing the Workbench...

- On 15/16/24 Bit Workbenches you cannot use 8 Bit patterns anymore. I have disabled the datatypes loader to have an easy test control over my loader. This will change in the future

1.17 rights

1.18 distrib

1.19 features

- NewPatterns speeds up the Workbench:

- Window Movement; Windows can be moved faster

- Window Sizing; Windows can be sized faster

- Window Opening/Closing; Windows appear/disappear much faster

- MagicLayers and clones run more faster now

- Images can be remapped with user defined precision

- All images look much better anyhow; it seems that the Workbench only used a fixed set of pens.

- Memory access to slower graphics boards can be optimized

- It is the only way to get REAL 15/16/24 Bit backdrops ;-)