ManyThings, 3.1 **9-16-94**

This program is for demo purposes only, I do not want you to send me any money. I just want to share a program I had fun writing. The original program was fun for me in that it was possible to do a windows 3.1 screen saver entirely in visual basic and windows calls without additional DLLs required. This latest version of the program now uses a DLL written in Borland C++ 3.1 to load DIB and GIF files into memory, do palette manipulations and write DIB format files to disk. This became necessary because Visual Basic was too slow to handle this.

Installation -- The program requires VBRUN300.DLL to be in your "\windows\system" directory (can be downloaded from many places), and of course Windows 3.1. You need to copy MANYTHNG.SCR (this is the executable) into your "\windows" directory and MNYTHDLL.DLL into your "\windows\system" directory. At this point you can select "Control Panel" from the "Main" group in program manager, and then from "Control Panel" select "Desktop". In desktop there is a section called Screen Saver. In that section under name you then push the down arrow and select "Many Things". At this point you sould probably click on the "Setup Button", this is where you may set the various options and can set a password if you wish. You may now test the saver by clicking "Test" and may set the time before windows will activate the saver by changing the "Delay" time. When you select "OK" from desktop, "Many Things" becomes the screen saver that windows will use.

Disclaimers: The program is provided as a programming example at no charge. The program is provided as is, without any warranties or obligations.

Short History --
New with version 3.1 -- Fixes problem if bitmaps directory is invalid.

New with version 3.0 -- Now implementing password protection! There is also a new ShootHoles saver, some minor changes of kaliedescope savers, and the line drawing color selection has been improved for 256 color displays so that greys are rejected. Also, it is now possible to change the frequency of individual savers, or even disabling certain savers by adding lines to "Control.ini". Also there is a color cycling saver that works with DIB and GIF-87a (non-interlaced) format files with both 256 color and Hi-Color displays. The slide show routine now also works with GIF-87a (non-interlaced) format files which are usually much smaller than BMP format files. These files may be converted using Paintshop Pro (highly recommended shareware) make sure to select format GIF and 87a -- non-interlaced. Warning -- interlaced GIF files will look like the picture is stacked on itself several times and each version is squashed; GIF-89a files will not load, they can be converted using Paintshop Pro. The C code for reading the GIF format was written by Larry Widing and is available in the Borland C++ Win/OS2 Forum, Library 10 area on compuserve as bitmaps.zip.

This is a screen saver was originally written in visual basic 2.0 (standard edition) and now has been upgraded to VB 3.0. The program actually cycles between several different screen savers within the same program to give quite a bit of variety. The basic screen saver method is based on an example in "Learn Programming and Visual Basic 2.0" by John Socha and Sybex Inc. The Windows API calls were chosen with help from PC Mag's "Visual Basic Programmer's Guide to the Windows API". The grabbing of the desktop pixels was based on Jonathan Zuck's screen-capture utilities in the November issue of Windows Tech Journal. Also thanks to Rick Perrott for suggestions on using setting a modal window and disabling windows screen save (please send me a note on how to contact you).

Screen Saver Operation --
When the saver begin the subroutine main in MANYTHNG.BAS is first called, it then reads CONTROL.INI and checks the command line parameters to see whether to run the setup form or to start the saver. When it starts the saver, it executes ManyThngs.Show which causes subroutine Form_Load to run in form MANYTHNG.FRM. This subroutine initializes variables, etc. Then from that point on everything is initiated by the timer. Whenever the timer times out, the subroutine Tick_Timer is called which handles the housekeeping for which save is to be run, and when it is time to change to another saver. Using the Tick_Timer means that the savers are limited for speed, but the will yield time to background processes (e.g. file transfers, tape backups, etc.).

The savers that involve moving lines or objects work by starting at random positions on the screen with zero velocity and acceleration vectors. Then in each iteration, a random acceleration is added to the velocity of each point. When an object gets to the edge of the screen the velocity's sign is reversed to bring it back into screen. This effect looks to the user as if the object bounced off the edge of the screen. Also when the velocity reaches a certain limit, the velocity is set to zero to prevent things from getting too out of hand.

Design Enviroment --
This program was originally developed on a 386/40MHz clone with 8MB and a JDR VGA 1024+ video card (ET4000 based) using Visual Basic 2.0. This was later upgraded to a 486/66MHz motherboard with 8MB and a ProLogic video card (true color) and Visual Basic 3.0. It has also been tested on an AST 486/33MHz system with video built into the motherboard. I am interested to hear about how the program works on your system, but please include details such as type of system, video card, display mode (e.g. 1024x768x256), amount of memory, and network stuff.

Features:

Version 3.0 -- Password protection is now available by a button click in the setup program. As with any password protection for a screen saver, be careful not to loose the password. If you do you can delete the password by editing "control.ini" in windows directory: in section "[Screen Saver.Many Things]". You can simply delete the line with "Password=". This program has the same password limitations as with any screen saver:

1. It does not prevent completely rebooting the PC.
2. Password protection can be defeated by editing "control.ini".
3. Do not forget your password or you will have to do steps 1 and 2.

The screen saver password protection only stops casual users from bothering with your PC. It will also prevent other users from accessing your network files if those are also password protected. There are still problems with stay on top tool bars, these cannot be drawn over. For reading the password while the saver is running, it was necessary to set "KeyPreview = True" along with calling "SetSysModalWindow". It was then necessary to use the KeyDown subroutine for other program's annoying pop-up dialogs would steal the keys from the KeyPress subroutine. Using KeyDown makes it necessary to convert the key codes to ascii codes using "MapVirtualKey". When the password protection is enabled and the mouse is moved or a key is pressed, then a password input box pops

up. It is not protected from the screen saver so it may get covered up, but if you continue to type or move the mouse it will get refreshed on the screen.

This saver uses a DLL written in Borland C++ 3.1 (source and dll is included). One feature is to emulate the palette cycling of FRACTINT (a freeware fractal generation program available for download) but also to work on a machine that was not running in palette mode (e.g. 64K colors). FRACTINT saves the fractals it generates as GIF-87a format which is now supported by the ManyThings screen saver. On a palette display the program will remap the palette entries, but on the high color displays it will load the image into memory and then repeatedly change the palette in the image and copy the image to the screen (this can flicker quite a bit on slow graphics cards). The ManyThings DLL includes routines to read DIB and GIF format files into a DIB format in memory, a routine to write the DIB in memory to a file, a routine for cycling the palette of the DIB in memory, routines to get pointers to the palette and data areas of the DIB, and a routine to free the memory used by the DIB.

To disable certain set a line in file "control.ini" in the windows directory under section "[Screen Saver.Many Things]". For example to disable saver number 20 put in line "Priority20=0" or "Priority1=0" to disable saver number 1. To change how frequently a saver comes up, this priority line can be set to other values. The default priority is 1.0, so for a saver to come up half as often as a saver with priority 1.0, set it to 0.5. Likewise setting it to 2.0 will make a saver come up twice as often as one with priority of 1.0.

Version 2.0 -- The program has been greatly re-written as I learned more about the Windows and Visual Basic environments. Each saver is in a separate subroutine and takes care of it's own initialization (when PlotInit is true) and its own cleanup before the next saver is called (when both PlotInit and PlotEnd are true, which resets the environment to default state and frees array memory). The program also has a trace mode which saves all the events when the program is running into a log file "c:\manythng.log". To enable, edit "control.ini" in windows directory: in section "[Screen Saver.Many Things]", add line: "ErrorTrace=ON". And to disable, delete that line or put a # in front of it. Using the trace mode helps quite a bit when implementing a new saver routine and problems are occuring. Also to test program from within Visual Basic set command line to "/t /s" using OPTIONS and PROJECT. This leaves the cursor on the screen and does not set system modal window so it is possible to single step through program after a breakpoint is reached. To configure the saver within Visual Basic, use "/c" on command line.
To add your own savers, you can add a subroutine to MANYTHNG.FRM (use the other saver routines as examples) and add a call to it to the switch statement in RunSelection. Then you need to increment the value of MaxPlotType in subroutine Form_Load.

Screen Saver Provided:

Number	Subroutine	Description
1	Squiggles	multiple scribbles wandering across the screen
2	Kalied2	draws lines which are mirrored, when maximum reached, screen cleared
3	Polygons	multiple sided objects wandering across screen with old objects erased

4	Circles	circles wandering across screen with old circles erased
5	Kalied	draws lines which are mirrored with old circles erased
6	Lines	multiple lines wandering across screen with old lines erased
7	Roll	screen rolls vertically or horizontally
8	FilledCircles	filled circles wandering across screen
9	Patch	patches of screen pasted at random positions on screen
10	Spiro	draws spirographs
11	Scrape	rectangle wanders across screen doing a random bitblit function
12	Stretch	screen is streched (like a zoom effect). Only works in 16 or 24 bit display modes.
13	Dribble	droplets are dribbled on the screen
14	Drop	strips of the screen fall to the bottom
15	Slides	does a slide show, randomly cycling through bitmaps and gifs in bitmap directory
16	FilledPolygons	filled polygons wandering across screen
17	MultiSpiros	draws multiple identical spirographs
18	Puzzle	divides screen into squares a jumbles them by shifting rows or columns by one position
19	ShootHoles	puts filled circles in a circular pattern on the screen using different coloring schemes
20	CyclePalette	reads any DIB or GIF-87a non-interlaced format files in the palette cycle directory and does a rapid cycling of the colors
21	Confettee	multiple dots of random color covering the screen

Feel free to experiment with the configuration settings for the saver. Those with only 4MB of memory and are running display modes that require a lot of memory to store screens, should probably select the low memory mode to reduce disk swapping. This disables the savers that save copies of the screen to modify. To disable the slide show, set bitmap directory to directory that does not have bitmaps or to invalid directory.

Lessons Learned:

Application Title must start with 'SCRNSAVE' and only some limited number of characters are recognized.

I was using a VB clock program that while minimized would change caption to show time. For some reason changing the caption would prevent windows from calling the screen saver.

In the windows desktop, I set timeout interval before saver is called to 1 minute for testing. I found that the display got slower and slower. Then when I would move the mouse, the display would change many times before returning to windows. It seems that windows

was calling another instance of the program every minute. This was because the saver needs to disable Windows from calling additional screen savers after the saver starts. This is done with a call to SetSysModalWindow.

PS -- even though I have Borland C++ and am a C++ programmer, I much prefer the Visual Basic development environment.

Bruce McLean
800 S. HW. 1417 #1214
Sherman TX 75090
CIS: 71413,2664
Internet: MCLB@TIMSG.CSC.TI.COM