ManyThings, 2.0 **2-17-94**

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. This 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. The downside of this is that Password protection is not included. The program requires VBRUN300.DLL, and of course Windows 3.1. You need to copy MANYTHNG.SCR into your windows directory.

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 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).

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.

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.

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). Plus each routine has a priority which it returns when PlotInit is true, PlotEnd is False, and PlotPriority is True. Increasing this value in a saver will mean that that saver will run more frequently (Priority/(Sum of Priorities), or by lowering it will run less frequently (only when sequence is random). 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.

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

Savers:

Number	Subroutine	Description
1	Squiggles	multiple scribbles wandering across the screen
2	Kalied2	draws lines which are mirrored, when maximum reached, screen is 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 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

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 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 for forms, debugging, etc.

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