Extra Credit Assignment – **C# Screen Saver**
GUI Programming
%1 added to final grade

A Windows screen saver is simply a Windows application that displays either an options dialog box, a preview screen, or a full-screen window, depending on the command line argument it receives. Windows screen savers have a .scr extension and are typically stored in the Windows system directory.

Argument	Action
/c	Configuration mode
/p	Preview mode
/s	Full-screen mode

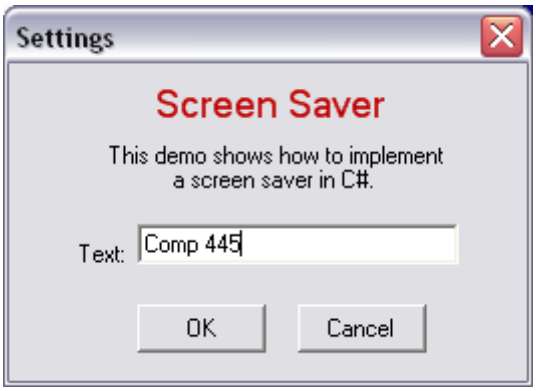
Your program can determine which mode your program is to execute by examining the command line parameters in Main:

```
static void Main()
{
        string[] cmdList = Environment.CommandLine.Split(' ');

        // First command-line param is the exe name, second param is the mode

        if (cmdList.Length >= 2) {
                if (cmdList[1].IndexOf("/c") >= 0) {        // Configuration mode
                        Application.Run(new OptionsForm());
                        return;
                }

                parentHwnd = IntPtr.Zero;

                if (cmdList[1] == "/p") {                   // Preview mode
                        previewMode = true;

                        // Handle to preview dialog box is next command-line param
                        parentHwnd = (IntPtr) uint.Parse(cmdList[2]);
                }
        }

        Application.Run(new ScreenSaverForm());
}
```

The parentHwnd will be used in preview mode and is discussed in the preview mode section. The ScreenSaverForm is a Windows Form that will need to fill the entire screen. This is done by setting the ScreenSaverForm's **WindowState** property to Maximized and **FormBorderStyle** to None. The OptionsForm is just a Windows Form that looks like a typical dialog box for setting properties of the screen saver.

Configuration Mode

In configuration mode, the screen saver should launch a dialog box that allows the user to control the appearance and settings of the screen saver. Typically these settings are stored in the Windows Registry. The Registry is a hierarchical repository of information used by Windows and applications for storing various types of configuration data.



The following code segment shows how information can be stored in and retrieved from the Registry:

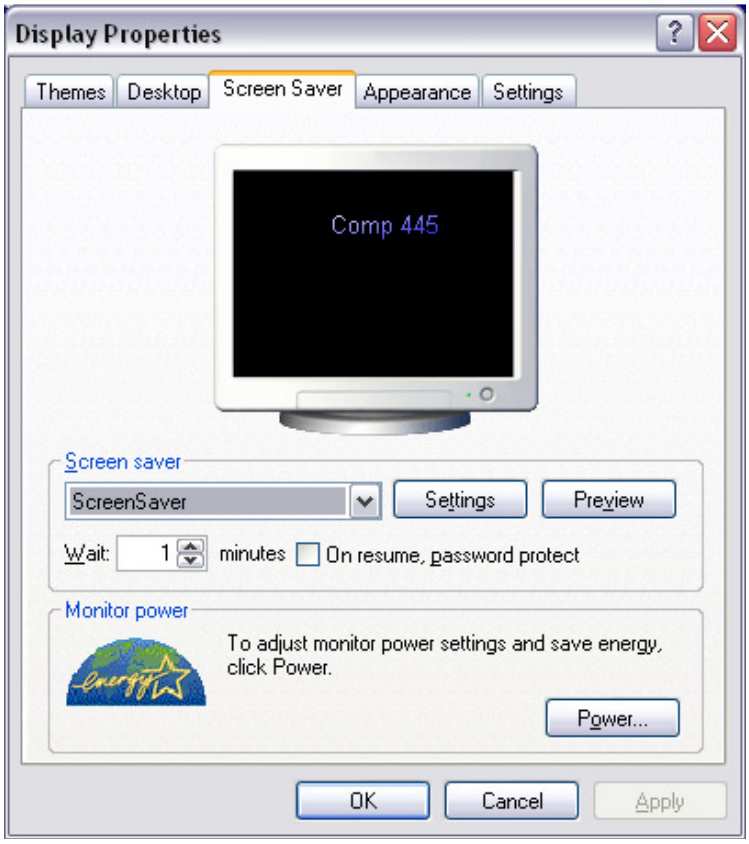
```
using Microsoft.Win32;

// Store in Registry
RegistryKey key = Registry.LocalMachine.OpenSubKey("SOFTWARE\\Demo_ScreenSaver");
if (key != null)
        Registry.LocalMachine.DeleteSubKeyTree("SOFTWARE\\Demo_ScreenSaver");
key = Registry.LocalMachine.CreateSubKey("SOFTWARE\\Demo_ScreenSaver");
key.SetValue("text", "some text");

// Read from Registry
RegistryKey key = Registry.LocalMachine.OpenSubKey("SOFTWARE\\Demo_ScreenSaver");
if (key != null)
        string text = (string)key.GetValue("text");
```

You can see the entries made in the Registry by using regedit. Run regedit by clicking on the Start button in the task bar, selecting Run… and entering "regedit". **By very careful when using regedit… deleting or altering any information may cause Windows or other applications not to function properly anymore.**

Look under "My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Demo_ScreenSaver" to see your setting. Close regedit when you no longer need it.

Preview Mode

When the screen saver starts up in preview mode, it should run in a child window whose parent window is the Display Property window as shown in the figure below. The handle to the Display Properties window is obtained from the second command line parameter when the screen saver is executed in preview mode.



In this mode, you will find the IsWindowVisible and GetClientRect Win32 API functions useful for determining if the Displays Property dialog is still visible and for determining the dimensions of the area you can paint to.

```
// Quit if dialog is dismissed.  Check this periodically.
if (!IsWindowVisible(parentHwnd))
        Application.Exit();

private struct RECT {
        public int left, top, right, bottom;
}

RECT rect = new RECT();
if (GetClientRect(parentHwnd, ref rect)) {
        Graphics g = Graphics.FromHwnd(parentHwnd);
        g.Clear(Color.Black);
        graphics.DrawString("Demo Screen Saver", font, brush, ((float)rect.right - 50),
                ((float)rect.bottom - 50);
        g.Dispose();
}
```

Full Screen Mode

In full screen mode, your application should run until it receives a mouse move, mouse click, or some other event that indicates it should terminate. The cursor should be hidden by calling `Cursor.Hide();`

Demo

The example C# application located at \\cs1\Classes\Comp445\C#\ScreenSaver shows a simple screen saver that displays text at a random location for a fixed period of time. Load this project into Visual Studio .NET and build the ScreenSaver.exe. Then change the extension of the .exe to .scr and copy it to the Windows\System32 directory where all the other screen savers are located.

Launch the Display Property dialog by right-clicking in the Windows desktop and selecting Properties from the pop-up menu. Select the Screen Saver tab located at the top of the dialog box and select “ScreenSaver” from the list of screen savers. You should see it running in preview mode in the small rectangular area of the dialog. Change the text being displayed by clicking the Settings button. Then view it full-screen by clicking the Preview button. Clicking the mouse button when the screen saver is running in full-screen mode will cause it to terminate.

Animation

Modify the screen saver to display some type of animation using any technique you’d like. Maybe you’d like to display a bouncing ball or a Harding Bison that moves across the screen. E-mail me your finished project files all zipped into a zip file.

Your project is due by Friday before finals week.

Do it big, do it right, and do it with style.

-Fred Astaire