

```

public class GraphicsDeviceManager : IGraphicsDeviceService, IDisposable, IGraphicsDeviceManager
{
    // Fields
    private bool allowMultiSampling;
    private SurfaceFormat backBufferFormat;
    private int backBufferHeight;
    private int backBufferWidth;
    private bool beginDrawOk;
    public static readonly int DefaultBackBufferHeight;
    public static readonly int DefaultBackBufferWidth;
    private static DepthFormat[] depthFormatsWithoutStencil;
    private static DepthFormat[] depthFormatsWithStencil;
    private DepthFormat depthStencilFormat;
    private GraphicsDevice device;
    private EventHandler deviceCreated;
    private EventHandler deviceDisposing;
    private static readonly TimeSpan deviceLostSleepTime;
    private EventHandler deviceReset;
    private EventHandler deviceResetting;
    private EventHandler Disposed;
    private Game game;
    private bool inDeviceTransition;
    private bool isDeviceDirty;
    private bool isFullScreen;
    private bool isReallyFullScreen;
    private ShaderProfile minimumPixelShaderProfile;
    private ShaderProfile minimumVertexShaderProfile;
    private static MultiSampleType[] multiSampleTypes;
    private EventHandler<PreparingDeviceSettingsEventArgs> PreparingDeviceSettings;
    private int resizedBackBufferHeight;
    private int resizedBackBufferWidth;
    private bool synchronizeWithVerticalRetrace;
    private bool useResizedBackBuffer;
    public static readonly SurfaceFormat[] ValidAdapterFormats;
    public static readonly SurfaceFormat[] ValidBackBufferFormats;
    public static readonly DeviceType[] ValidDeviceTypes;

    // Events
    public event EventHandler DeviceCreated;
    public event EventHandler DeviceDisposing;
    public event EventHandler DeviceReset;
    public event EventHandler DeviceResetting;
    public event EventHandler Disposed;
    public event EventHandler<PreparingDeviceSettingsEventArgs> PreparingDeviceSettings;

    // Methods
    static GraphicsDeviceManager();
    public GraphicsDeviceManager(Game game);
    private void AddDevices(bool anySuitableDevice, List<GraphicsDeviceInformation> foundDevices);
    private void AddDevices(GraphicsAdapter adapter, DeviceType deviceType, DisplayMode mode, GraphicsDeviceInformation baseDeviceInfo, List<GraphicsDeviceInformation> foundDevices);
    public void ApplyChanges();
    protected virtual bool CanResetDevice(GraphicsDeviceInformation newDeviceInfo);
    private void ChangeDevice(bool forceCreate);
    private void CheckForAvailableSupportedHardware();
    private DepthFormat ChooseDepthStencilFormat(GraphicsAdapter adapter, DeviceType deviceType, SurfaceFormat adapterFormat);
    private DepthFormat ChooseDepthStencilFormatFromList(DepthFormat[] availableFormats, GraphicsAdapter adapter, DeviceType deviceType, SurfaceFormat adapterFormat);
    private void CreateDevice(GraphicsDeviceInformation newInfo);
    private Exception CreateNoSuitableGraphicsDeviceException(string message, Exception innerException);
    protected virtual void Dispose(bool disposing);
    private bool EnsureDevice();
    private bool EnsureDevicePlatform();
    protected virtual GraphicsDeviceInformation FindBestDevice(bool anySuitableDevice);
    private GraphicsDeviceInformation FindBestPlatformDevice(bool anySuitableDevice);
    private void GameWindowClientSizeChanged(object sender, EventArgs e);
    private void GameWindowScreenDeviceNameChanged(object sender, EventArgs e);
    [DllImport("user32.dll")]
    private static extern int GetSystemMetrics(uint smIndex);
    private void HandleDeviceLost(object sender, EventArgs e);
    private void HandleDeviceReset(object sender, EventArgs e);
    private void HandleDeviceResetting(object sender, EventArgs e);
    private void HandleDisposing(object sender, EventArgs e);
    private static bool IsValidShaderProfile(ShaderProfile capsShaderProfile, ShaderProfile minimumShaderProfile);
    private bool IsWindowOnAdapter(IntPtr windowHandle, GraphicsAdapter adapter);
    private void MessagePresentParameters(PresentationParameters pp);
    bool IGraphicsDeviceManager.BeginDraw();
    void IGraphicsDeviceManager.CreateDevice();
    void IGraphicsDeviceManager.EndDraw();
    protected virtual void OnDeviceCreated(object sender, EventArgs args);
    protected virtual void OnDeviceDisposing(object sender, EventArgs args);
    protected virtual void OnDeviceReset(object sender, EventArgs args);
    protected virtual void OnDeviceResetting(object sender, EventArgs args);
    protected virtual void OnPreparingDeviceSettings(object sender, PreparingDeviceSettingsEventArgs args);
    protected virtual void RankDevices(List<GraphicsDeviceInformation> foundDevices);
    private void RankDevicesPlatform(List<GraphicsDeviceInformation> foundDevices);
    void IDisposable.Dispose();
    public void ToggleFullScreen();
    private void ValidateGraphicsDeviceInformation(GraphicsDeviceInformation devInfo);

    // Properties
    public GraphicsDevice GraphicsDevice { get; }
    public bool IsFullScreen { get; set; }
    public ShaderProfile MinimumPixelShaderProfile { get; set; }
    public ShaderProfile MinimumVertexShaderProfile { get; set; }
    public bool PreferMultiSampling { get; set; }
    public SurfaceFormat PreferredBackBufferFormat { get; set; }
    public int PreferredBackBufferHeight { get; set; }
    public int PreferredBackBufferWidth { get; set; }
    public DepthFormat PreferredDepthStencilFormat { get; set; }
    public bool SynchronizeWithVerticalRetrace { get; set; }
}

```

Expand Methods