

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
using System.Runtime.InteropServices;
using System.Windows.Forms;

namespace sb54_CSAI.GUI
{
    #region ShrinkPanel CLASS
    /// <summary>
    /// ShrinkPanel is a sub class of a <see cref="System.Windows.Forms.Panel">Panel </see>
    /// that provides Shrink Panels like those provided in Windows XP.
    /// <br></br>
    /// <br></br>
    /// This class was written using an article contained at the following URL
    /// http://www.codeproject.com/cs/miscctrl/CollapsiblePanelBar.asp?df=100&forumid=12638&exp=0&fr=51
    /// <br></br>
    /// GUI components are an area that interest me so I have included the source code in
    this namespace so that
    /// I could understand what was being done by the Author (Derek Lakin) of the original
    classes.
    /// I could have simply used the Authors original C# Dll, but I would not have learned
    much that way
    /// so I chose to examine and use the source code. Some of the names of classes are
    different here, as I
    /// actually wrote the code line by line, using the original article so that I could
    fully comprehend the
    /// original implementation. As such all the comments that are found within the code are
    by and large my own
    /// comments
    /// </summary>
    public class ShrinkPanel : System.Windows.Forms.Panel
    {
        #region Instance Fields
        //Instance Fields
        private ColorMatrix disabledMatrix;
        private ImageAttributes disabledAttributes;
        private ShrinkPanelCurrentState state = ShrinkPanelCurrentState.Expanded;
        private int oPanelHeight;
        private int imageIndex = 0;
        private const int minTitleHeight = 24;
        private const int iconBorder = 2;
        private const int expandBorder = 4;
        private Color startColour = Color.White;
        private Color endColour = Color.FromArgb(199, 212, 247);
        private IContainer components;
        private Label labelTitle;
        private Image image;
        private ImageList imageList;
        /// <summary>
        /// A ShrinkPanelCurrentState changed event.
        /// </summary>
        [Category("State"),
        Description("Raised when Panel state has changed.")]
        public event ShrinkPanelCurrentStateChangedEventHandler
        ShrinkPanelCurrentStateChanged;
        #endregion
        #region Public Constructors
        /// <summary>
        /// Initialises a new instance of <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel
        </see>
        /// This constructor inherits from the <see cref="System.Windows.Forms.Panel">Panel
        </see>
        /// constructor
        /// </summary>
        public ShrinkPanel() : base()
        {
```

```

        //get new components container
        this.components = new System.ComponentModel.Container();
        //create GUI components
        InitializeComponent();
        // Set the background colour to ControlLightLight
        this.BackColor = Color.AliceBlue;
        // Store the current oPanelHeight
        this.oPanelHeight = this.Height;
        // Setup the ColorMatrix and ImageAttributes for grayscale (disabled) images.
        this.disabledMatrix = new ColorMatrix();
        this.disabledMatrix.Matrix00 = 1/3f;
        this.disabledMatrix.Matrix01 = 1/3f;
        this.disabledMatrix.Matrix02 = 1/3f;
        this.disabledMatrix.Matrix10 = 1/3f;
        this.disabledMatrix.Matrix11 = 1/3f;
        this.disabledMatrix.Matrix12 = 1/3f;
        this.disabledMatrix.Matrix20 = 1/3f;
        this.disabledMatrix.Matrix21 = 1/3f;
        this.disabledMatrix.Matrix22 = 1/3f;
        this.disabledAttributes = new ImageAttributes();
        this.disabledAttributes.SetColorMatrix(this.disabledMatrix, ColorMatrixFlag.
Default,
        ColorAdjustType.Bitmap);
    }
    #endregion
    #region Public Methods/Properties
    /// <summary>
    /// Gets/sets the <see cref="sb54_CSAI.GUI.ShrinkPanelCurrentState">
    ShrinkPanelCurrentState</see>, which
    /// can either be Expanded/Collapsed
    /// </summary>
    [Browsable(false)]
    public ShrinkPanelCurrentState ShrinkPanelCurrentState
    {
        get
        {
            return this.state;
        }
        set
        {
            ShrinkPanelCurrentState oldState = this.state;
            this.state = value;
            if(oldState != this.state)
            {
                // State has changed to update the display
                UpdateDisplayedState();
            }
        }
    }

    /// <summary>
    /// Gets/sets the text displayed as the oPanel title.
    /// </summary>
    [Category("Title"),
    Description("The text contained in the title bar.")]
    public string TitleText
    {
        get
        {
            return this.labelTitle.Text;
        }
        set
        {
            this.labelTitle.Text = value;
        }
    }

    /// <summary>
    /// Gets/sets the foreground colour used for the title bar.
    /// </summary>
    [Category("Title"),
    Description("The foreground colour used to display the title text.")]
    public Color TitleFontColour

```

```
{
    get
    {
        return this.labelTitle.ForeColor;
    }
    set
    {
        this.labelTitle.ForeColor = value;
    }
}

/// <summary>
/// Gets/sets the font used for the title bar text.
/// </summary>
[Category("Title"),
Description("The font used to display the title text.")]
public Font TitleFont
{
    get
    {
        return this.labelTitle.Font;
    }
    set
    {
        this.labelTitle.Font = value;
    }
}

/// <summary>
/// Gets/sets the image list used for the expand/collapse image.
/// </summary>
[Category("Title"),
Description("The image list to get the images displayed for expanding/collapsing the
oPanel.")]
public ImageList ImageList
{
    get
    {
        return this.imageList;
    }
    set
    {
        this.imageList = value;
        //is the imageList valid
        if(this.imageList != null)
        {
            //are there images within it
            if(this.imageList.Images.Count > 0)
            {
                this.imageIndex = 0;
            }
        }
        else
        {
            this.imageIndex = -1;
        }
    }
}

/// <summary>
/// Gets/sets the starting colour for the background gradient of the header.
/// </summary>
[Category("Title"),
Description("The colour used at the start of the colour gradient displayed as the
background of the title bar.")]
public Color StartColour
{
    get
    {
        return this.startColour;
    }
    set
    {
```

```

        this.startColour = value;
        this.labelTitle.Invalidate();
    }
}

/// <summary>
/// Gets/sets the ending colour for the background gradient of the header.
/// </summary>
[Category("Title"),
Description("The colour used at the end of the colour gradient displayed as the
background of the title bar.")]
public Color EndColour
{
    get
    {
        return this.endColour;
    }
    set
    {
        this.endColour = value;
        this.labelTitle.Invalidate();
    }
}

/// <summary>
/// Gets/sets the image displayed in the header of the title bar.
/// </summary>
[Category("Title"),
Description("The image that will be displayed on the left hand side of the title bar
.")]
public Image Image
{
    get
    {
        return this.image;
    }
    set
    {
        this.image = value;
        if(null != value)
        {
            // Update the height of the title label
            this.labelTitle.Height = this.image.Height + (2 * ShrinkPanel.
iconBorder);
            if(this.labelTitle.Height < minTitleHeight)
            {
                this.labelTitle.Height = minTitleHeight;
            }
            this.labelTitle.Invalidate();
        }
    }
}
#endregion
#region Protected Methods
/// <summary>
/// Raises the OnShrinkPanelCurrentStateChanged event for the <see cref="sb54_CSAI.
GUI.ShrinkPanel.ShrinkPanelCurrentStateChanged">ShrinkPanelCurrentStateChanged</see>.
/// Any users of the <see cref="sb54_CSAI.GUI.ShrinkPanel.
ShrinkPanelCurrentStateChanged">ShrinkPanelCurrentStateChanged</see> may now subscribe
to this event
/// using a ShrinkPanelCurrentStateChangedEventHandler delegate.
/// </summary>
/// <param name="e">The <see cref="sb54_CSAI.GUI.PanelEventArgs">PanelEventArgs</
see> that contains the event data</param>
protected virtual void OnShrinkPanelCurrentStateChanged(PanelEventArgs e)
{
    if(ShrinkPanelCurrentStateChanged != null)
    {
        // Invokes the delegates.
        ShrinkPanelCurrentStateChanged(this, e);
    }
}
#endregion

```

```

#region Component Designer generated code
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.components = new System.ComponentModel.Container();
    System.Resources.ResourceManager resources = new System.Resources.ResourceManager(
ResourceManager(typeof(ShrinkPanel));
    this.labelTitle = new System.Windows.Forms.Label();
    this.imageList = new System.Windows.Forms.ImageList(this.components);
    this.SuspendLayout();
    //
    // labelTitle
    //
    this.labelTitle.Cursor = System.Windows.Forms.Cursors.Default;
    this.labelTitle.Dock = System.Windows.Forms.DockStyle.Top;
    this.labelTitle.Font = new System.Drawing.Font("Tahoma", 9F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.labelTitle.ForeColor = System.Drawing.Color.Navy;
    this.labelTitle.Location = new System.Drawing.Point(114, 17);
    this.labelTitle.Name = "labelTitle";
    this.labelTitle.Size = new System.Drawing.Size(200, 24);
    this.labelTitle.TabIndex = 0;
    this.labelTitle.Text = "Title";
    this.labelTitle.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
    this.labelTitle.Paint += new System.Windows.Forms.PaintEventHandler(this.labelTitle_Paint);
    this.labelTitle.MouseUp += new System.Windows.Forms.MouseEventHandler(this.labelTitle_MouseUp);
    this.labelTitle.MouseMove += new System.Windows.Forms.MouseEventHandler(this.labelTitle_MouseMove);
    //
    // imageList
    //
    this.imageList.ColorDepth = System.Windows.Forms.ColorDepth.Depth32Bit;
    this.imageList.ImageSize = new System.Drawing.Size(16, 16);
    this.imageList.ImageStream = ((System.Windows.Forms.ImageListStreamer)(resources.GetObject("imageList.ImageStream",
System.Globalization.CultureInfo.InvariantCulture)));
    this.imageList.TransparentColor = System.Drawing.Color.Transparent;
    //
    // ShrinkPanel
    //
    this.Controls.AddRange(new System.Windows.Forms.Control[] {
this.labelTitle});
    this.ResumeLayout(false);
}
#endregion
#region Private Methods
#region Helper functions
/// <summary>
/// Helper function to determine if the mouse is currently over the title bar.
/// </summary>
/// <param name="xPos">The x-coordinate of the mouse position.</param>
/// <param name="yPos">The y-coordinate of the mouse position.</param>
/// <returns></returns>
private bool IsOverTitle(int xPos, int yPos)
{
    // Get the dimensions of the title label
    Rectangle rectTitle = this.labelTitle.Bounds;
    // Check if the supplied coordinates are over the title label
    if(rectTitle.Contains(xPos, yPos))
    {
        return true;
    }
    return false;
}

/// <summary>
/// Helper function to update the displayed state of the oPanel.

```

```

/// </summary>
private void UpdateDisplayedState()
{
    switch(this.state)
    {
        case ShrinkPanelCurrentState.Collapsed :
            // Entering collapsed state, so store the current height.
            this.oPanelHeight = this.Height;
            // Collapse the oPanel
            this.Height = labelTitle.Height;
            // Update the image.
            this.imageIndex = 1;
            break;
        case ShrinkPanelCurrentState.Expanded :
            // Entering expanded state, so expand the oPanel.
            this.Height = this.oPanelHeight;
            // Update the image.
            this.imageIndex = 0;
            break;
        default :
            // Ignore
            break;
    }
    this.labelTitle.Invalidate();

    OnShrinkPanelCurrentStateChanged(new PanelEventArgs(this));
}
#endregion
#region GUI Look And Feel Methods
/// <summary>
/// Paint the main title area using a color blend and draws the icon and Expand/
Collapse areas
/// </summary>
/// <param name="sender">The source object that raised the event</param>
/// <param name="e">The event arguments</param>
private void labelTitle_Paint(object sender, System.Windows.Forms.PaintEventArgs e)
{
    const int diameter = 14;
    int radius = diameter / 2;
    Rectangle bounds = labelTitle.Bounds;
    int offsetY = 0;
    if(this.image != null)
    {
        offsetY = this.labelTitle.Height - ShrinkPanel.minTitleHeight;
        if(offsetY < 0)
        {
            offsetY = 0;
        }
        bounds.Offset(0, offsetY);
        bounds.Height -= offsetY;
    }
    //start with clear graphics object
    e.Graphics.Clear(this.Parent.BackColor);

    // Create a GraphicsPath with curved top corners
    GraphicsPath path = new GraphicsPath();
    path.AddLine(bounds.Left + radius, bounds.Top, bounds.Right - diameter - 1,
bounds.Top);
    path.AddArc(bounds.Right - diameter - 1, bounds.Top, diameter, diameter, 270,
90);
    path.AddLine(bounds.Right, bounds.Top + radius, bounds.Right, bounds.Bottom);
    path.AddLine(bounds.Right, bounds.Bottom, bounds.Left - 1, bounds.Bottom);
    path.AddArc(bounds.Left, bounds.Top, diameter, diameter, 180, 90);

    // Create a colour gradient
    e.Graphics.SmoothingMode = SmoothingMode.AntiAlias;
    if(this.Enabled)
    {
        LinearGradientBrush brush = new LinearGradientBrush(
            bounds, this.startColour, this.endColour, LinearGradientMode.Horizontal)
;

        // Paint the colour gradient into the title label.

```

```

        e.Graphics.FillPath(brush, path);
    }
    else
    {
        Colour grayStart = new Colour();
        grayStart.CurrentColour = this.startColour;
        grayStart.Saturation = 0f;
        Colour grayEnd = new Colour();
        grayEnd.CurrentColour = this.endColour;
        grayEnd.Saturation = 0f;
        LinearGradientBrush brush = new LinearGradientBrush(
            bounds, grayStart.CurrentColour, grayEnd.CurrentColour,
            LinearGradientMode.Horizontal);

        // Paint the grayscale gradient into the title label.
        e.Graphics.FillPath(brush, path);
    }

    // Draw the header icon, if there is one
    System.Drawing.GraphicsUnit graphicsUnit = System.Drawing.GraphicsUnit.Display;
    int offsetX = ShrinkPanel.iconBorder;
    if(this.image != null)
    {
        offsetX += this.image.Width + ShrinkPanel.iconBorder;
        //Draws the title icon grayscale when the oPanel is disabled.
        RectangleF srcRectF = this.image.GetBounds(ref graphicsUnit);
        Rectangle destRect = new Rectangle(ShrinkPanel.iconBorder,
            ShrinkPanel.iconBorder, this.image.Width, this.image.Height);
        if(this.Enabled)
        {
            e.Graphics.DrawImage(this.image, destRect, (int)srcRectF.Left, (int)
srcRectF.Top,
                (int)srcRectF.Width, (int)srcRectF.Height, graphicsUnit);
        }
        else
        {
            e.Graphics.DrawImage(this.image, destRect, (int)srcRectF.Left, (int)
srcRectF.Top,
                (int)srcRectF.Width, (int)srcRectF.Height, graphicsUnit, this.
disabledAttributes);
        }
    }

    // Draw the title text.
    SolidBrush textBrush = new SolidBrush(this.TitleFontColour);
    // Title text truncated with an ellipsis where necessary.
    float left = (float)offsetX;
    float top = (float)offsetY + (float)ShrinkPanel.expandBorder;
    float width = (float)this.labelTitle.Width - left - this.imageList.ImageSize.
Width -
        ShrinkPanel.expandBorder;
    float height = (float)ShrinkPanel.minTitleHeight - (2f * (float)ShrinkPanel.
expandBorder);
    RectangleF textRectF = new RectangleF(left, top, width, height);
    StringFormat format = new StringFormat();
    format.Trimming = StringTrimming.EllipsisWord;
    // Draw title text disabled where appropriate.
    if(true == this.Enabled)
    {
        e.Graphics.DrawString(labelTitle.Text, labelTitle.Font, textBrush,
            textRectF, format);
    }
    else
    {
        Color disabled = SystemColors.GrayText;
        ControlPaint.DrawStringDisabled(e.Graphics, labelTitle.Text, labelTitle.Font,
            disabled, textRectF, format);
    }

    // Draw a white line at the bottom:
    const int lineWidth = 1;
    SolidBrush lineBrush = new SolidBrush(Color.White);

```

```

    Pen linePen = new Pen(lineBrush, lineWidth);
    path.Reset();
    path.AddLine(bounds.Left, bounds.Bottom - lineWidth, bounds.Right,
        bounds.Bottom - lineWidth);
    e.Graphics.DrawPath(linePen, path);

    // Draw the expand/collapse image
    int xPos = bounds.Right - this.imageList.ImageSize.Width - ShrinkPanel.
expandBorder;
    int yPos = bounds.Top + ShrinkPanel.expandBorder;
    RectangleF srcIconRectF = this.ImageList.Images[(int)this.state].GetBounds(ref
graphicsUnit);
    Rectangle destIconRect = new Rectangle(xPos, yPos,
        this.imageList.ImageSize.Width, this.imageList.ImageSize.Height);
    if(true == this.Enabled)
    {
        e.Graphics.DrawImage(this.ImageList.Images[(int)this.state], destIconRect,
            (int)srcIconRectF.Left, (int)srcIconRectF.Top, (int)srcIconRectF.Width,
            (int)srcIconRectF.Height, graphicsUnit);
    }
    else
    {
        e.Graphics.DrawImage(this.ImageList.Images[(int)this.state], destIconRect,
            (int)srcIconRectF.Left, (int)srcIconRectF.Top, (int)srcIconRectF.Width,
            (int)srcIconRectF.Height, graphicsUnit, this.disabledAttributes);
    }
}
#endregion
#region GUI Interactions
/// <summary>
/// Changes the state of the <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see>
/// if the user clicks the title area of the panel
/// </summary>
/// <param name="sender">The source object that raised the event</param>
/// <param name="e">The event arguments</param>
private void labelTitle_MouseUp(object sender, System.Windows.Forms.MouseEventArgs
e)
{
    //was the click in the correct area to change the panel state
    if((e.Button == MouseButton.Left) && (IsOverTitle(e.X, e.Y)))
    {
        if((this.imageList != null) && (this.imageList.Images.Count >=2))
        {
            if(this.imageIndex ==0)
            {
                // Currently expanded, so store the current height.
                this.state = ShrinkPanelCurrentState.Collapsed;
            }
            else
            {
                // Currently collapsed, so expand the oPanel.
                this.state = ShrinkPanelCurrentState.Expanded;
            }
            UpdateDisplayedState();
        }
    }
}

/// <summary>
/// Change the cursor type dependant on the area that the user is hovering over with
/// the mouse
/// </summary>
/// <param name="sender">The source object that raised the event</param>
/// <param name="e">The event arguments</param>
private void labelTitle_MouseMove(object sender, System.Windows.Forms.MouseEventArgs
e)
{
    //Change the cursor type dependant on the area that the user is hovering over
with
    //the mouse
    if((e.Button == MouseButton.None) && (IsOverTitle(e.X, e.Y)))
    {
        this.labelTitle.Cursor = Cursors.Hand;
    }
}

```

```

    }
    else
    {
        this.labelTitle.Cursor = Cursors.Default;
    }
}
#endregion
#endregion
}
#endregion
#region Public Enumeration
/// <summary>
/// Defines the state of a <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see>.
/// </summary>
public enum ShrinkPanelCurrentState
{
    /// <summary>
    /// The <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see> is expanded.
    /// </summary>
    Expanded,
    /// <summary>
    /// The <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see> is collapsed.
    /// </summary>
    Collapsed
}
#endregion
#region Delegates
/// <summary>
/// A delegate type for hooking up a <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see> state
/// change notifications.
/// </summary>
public delegate void ShrinkPanelCurrentStateChangedEventHandler(object sender,
PanelEventArgs e);
#endregion
#region PanelEventArgs CLASS
/// <summary>
/// Provides PanelEventArgs for the <see cref="sb54_CSAI.GUI.ShrinkPanel.
ShrinkPanelCurrentStateChanged">ShrinkPanelCurrentStateChanged</see> event.
/// This event is then used by the <see cref="sb54_CSAI.GUI.ShrinkPanelBar">
ShrinkPanelBar</see> to determine the new layout
/// positions for all of the <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see>
contained.
/// <br></br>
/// <br></br>
/// This class was written using an article contained at the following URL
/// http://www.codeproject.com/cs/miscctrl/CollapsiblePanelBar.asp?df=100&forumid=12638&
exp=0&fr=51
/// <br></br>
/// GUI components are an area that interest me so I have written the classes in this
namespace so that
/// I could understand what was being done by the Author (Derek Lakin) of the original
classes. Some of the
/// names of classes are different here, as I actually wrote the code line by line,
using the original article
/// so that I could fully comprehend the original implementation. As such all the
comments that are found within the code are by and large my own
/// comments
/// </summary>
public class PanelEventArgs : System.EventArgs
{
    #region Instance Fields
    //Instance Fields
    private ShrinkPanel oPanel;
    #endregion
    #region Public Constructor
    /// <summary>
    /// Simply creates a new PanelEventArgs object using the parameter provided
    /// </summary>
    /// <param name="sender">The originating <see cref="sb54_CSAI.GUI.ShrinkPanel">
ShrinkPanel</see>.</param>
    public PanelEventArgs(ShrinkPanel sender)
    {

```

```
        this.oPanel = sender;
    }
#endregion
#region Public Properties
    /// <summary>
    /// Gets the <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see> that triggered
the event.
    /// </summary>
    public ShrinkPanel ShrinkPanel
    {
        get
        {
            return this.oPanel;
        }
    }

    /// <summary>
    /// Gets the ShrinkPanelCurrentState of the <see cref="sb54_CSAI.GUI.ShrinkPanel">
ShrinkPanel</see> that triggered the event.
    /// </summary>
    public ShrinkPanelCurrentState ShrinkPanelCurrentState
    {
        get
        {
            return this.oPanel.ShrinkPanelCurrentState;
        }
    }
#endregion
}
#endregion
}
```