

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

namespace sb54_CSAI.GUI
{
    #region ShrinkPanelBar CLASS
    /// <summary>
    /// uctCDRipper is a sub class of a <see cref="System.Windows.Forms.Panel">Panel </see>
    /// An ExplorerBar-type extended Panel for containing <see cref="sb54_CSAI.GUI.
    ShrinkPanel">ShrinkPanel</see> objects
    /// <br></br>
    /// <br></br>
    /// This code makes use of the <see cref="System.Windows.Forms.Panel">ISupportInitialize
    </see>interface
    /// to allow the the initilisation functions such as BeginInit, SuspendLayout,
    ResumeLayout to be used
    /// within the standard Windows Form Designer generated code
    /// <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 ShrinkPanelBar : System.Windows.Forms.Panel, System.ComponentModel.
    ISupportInitialize
    {
        #region Instance Fields
        //Instance Fields
        private ShrinkPanelCollection oPanels = new ShrinkPanelCollection();
        private int border = 8;
        private int spacing = 8;
        private bool initialising = false;
        #endregion
        #region Public Constructor
        /// <summary>
        /// Initialises a new instance of <see cref="sb54_CSAI.GUI.ShrinkPanelBar">
        ShrinkPanelBar</see>.
        /// </summary>
        public ShrinkPanelBar() : base()
        {
            InitializeComponent();

            //start with a default color
            this.BackColor = Color.CornflowerBlue;
        }
        #endregion
        #region Windows Form Designer generated code
        private void InitializeComponent()
        {
            //
            // ShrinkPanelBar
            //
        }
    }
}
```

```
#endregion
#region Public Properties / Methods
/// <summary>
/// Gets the <see cref="sb54_CSAI.GUI.ShrinkPanelCollection">ShrinkPanelCollection</see>
see> collection.
/// </summary>
[Browsable(false)]
public ShrinkPanelCollection ShrinkPanelCollection
{
    get
    {
        return this.oPanels;
    }
}

/// <summary>
/// Gets/sets the border around the oPanels.
/// </summary>
public int Border
{
    get
    {
        return this.border;
    }
    set
    {
        this.border = value;
        UpdatePositions(this.oPanels.Count - 1);
    }
}

/// <summary>
/// Gets/sets the vertical spacing between adjacent oPanels.
/// </summary>
public int Spacing
{
    get
    {
        return this.spacing;
    }
    set
    {
        this.spacing = value;
        UpdatePositions(this.oPanels.Count - 1);
    }
}

/// <summary>
/// Signals that initialization is starting.
/// </summary>
public void BeginInit()
{
    this.initialising = true;
}

/// <summary>
/// Signals that initialization is complete.
/// </summary>
public void EndInit()
{
    this.initialising = false;
}
#endregion
#region Protected Methods
/// <summary>
/// Event handler for the <see cref="System.Windows.Forms.Panel.Control.ControlAdded"
">ControlAdded</see> event.
/// </summary>
/// <param name="e">A <see cref="System.Windows.Forms.ControlEventArgs">
ControlEventArgs</see> that contains the event data.</param>
protected override void OnControlAdded(ControlEventArgs e)
{
    //call base (Panel)

```

```

        base.OnControlAdded(e);

        //is the event for the ShrinkPanel object type
        if(e.Control is ShrinkPanel)
        {
            // Adjust the docking property to Left | Right | Top
            e.Control.Anchor = AnchorStyles.Left | AnchorStyles.Top | AnchorStyles.Right;
        }
    }

    if(initialising)
    {
        // In the middle of InitializeComponent call.
        // Generated code adds oPanels in reverse order, so add to end
        this.oPanels.Add((ShrinkPanel)e.Control);
        //hook up event for panel change of state
        this.oPanels.Item(this.oPanels.Count - 1).ShrinkPanelCurrentStateChanged +=
            new ShrinkPanelCurrentStateChangedEventHandler(this.
panel_StateChanged);
    }
    else
    {
        // Add the panel to the beginning of the internal collection.
        oPanels.Insert(0, (ShrinkPanel)e.Control);
        //hook up event for panel change of state
        oPanels.Item(0).ShrinkPanelCurrentStateChanged +=
            new ShrinkPanelCurrentStateChangedEventHandler(this.
panel_StateChanged);
    }

    // Update the size and position of the oPanels
    UpdatePositions(this.oPanels.Count - 1);
}

/// <summary>
/// Event handler for the <see cref="System.Windows.Forms.Panel.Control.
ControlRemoved">ControlRemoved</see> event.
/// </summary>
/// <param name="e">A <see cref="System.Windows.Forms.ControlEventArgs">
ControlEventArgs</see> that contains the event data.</param>
protected override void OnControlRemoved(ControlEventArgs e)
{
    //call base (Panel)
    base.OnControlRemoved(e);

    //is the event for the ShrinkPanel object type
    if(e.Control is ShrinkPanel)
    {
        // Get the index of the panel within the collection.
        int index = this.oPanels.IndexOf((ShrinkPanel)e.Control);
        if(-1 != index)
        {
            // Remove this panel from the collection.
            this.oPanels.Remove(index);
            // Update the position of any remaining oPanels.
            UpdatePositions(this.oPanels.Count - 1);
        }
    }
}
#endregion
#region Private Methods
#region Private Helper Functions
/// <summary>
/// Updates all other panel positions based on the new layout required. Which is
dictated by
/// which one of the <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanels </see>just
shrunk/expanded
/// </summary>
/// <param name="index">The index of the panel within the internal collection of
/// <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanels </see>that has just shrunk/
expanded</param>
private void UpdatePositions(int index)

```

```

    {
        //loop through all panels contained within this object
        for(int i = index; i >= 0; i--)
        {
            // Update the panel locations.
            if(i == this.oPanels.Count - 1)
            {
                // Top panel.
                this.oPanels.Item(i).Top = this.border;
            }
            else
            {
                this.oPanels.Item(i).Top = this.oPanels.Item(i + 1).Bottom + this.border;
            }
            // Update the panel widths.
            this.oPanels.Item(i).Left = this.spacing;
            this.oPanels.Item(i).Width = this.Width - (2 * this.spacing);
            if(true == this.VScroll)
            {
                this.oPanels.Item(i).Width -= SystemInformation.VerticalScrollBarWidth;
            }
        }
    }
    #endregion
    #region GUI User interactions
    /// <summary>
    /// Event that occurs when one of the <see cref="sb54_CSAI.GUI.ShrinkPanel">
    ShrinkPanel</see>
    /// changes state. The other <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanels</
    see> will then
    /// need to be updated with new positions, so the UpdatePositions() method is called
    /// with the index of the <see cref="sb54_CSAI.GUI.ShrinkPanel">ShrinkPanel</see>
    that
    /// just changed state.
    /// </summary>
    /// <param name="sender">The source object that raised the event</param>
    /// <param name="e">The event arguments</param>
    private void panel_StateChanged(object sender, PanelEventArgs e)
    {
        // Get the index of the control that just changed state.
        int index = this.oPanels.IndexOf(e.ShrinkPanel);
        if(-1 != index)
        {
            // Now update the position of all subsequent oPanels.
            UpdatePositions(--index);
        }
    }
    #endregion
    #endregion
}
#endregion
}

```