

# Namespace ASE\_Assignment

## Classes

### [Draws](#)

This class extends the ICanvas Interface, it contains methods that allow initialization and usage of a bitmap to enable users to draw shapes.

### [Form1](#)

Main class of the program. It contains the graphical elements of the interface that the user sees alongside methods that initialize and handle user input

### [ownCommandfactory](#)

The class extends CommandFactory from BOOSE. Both classes receive commands from the user as input and create new objects of their corresponding classes, if they exist in the factory.

### [ownPenColour](#)

Class used to create a pen of a specified colour

# Class Draws

Namespace: [ASE Assignment](#)

Assembly: ASE Assignment.dll

This class extends the ICanvas Interface, it contains methods that allow initialization and usage of a bitmap to enable users to draw shapes.

```
public class Draws : ICanvas
```








## Inheritance

[object](#)  ← Draws

## Implements

ICanvas

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Remarks

The class implements pens, brushes, bitmaps and graphics that other parts of the program rely on. It is the class that handles what processed user input is drawn on.

## Constructors

### Draws()

Constructor for the Draws class, calls a Set function which initializes necessary components.

```
public Draws()
```

## Properties

### PenColour

penColour variable setter and getter

```
public object PenColour { get; set; }
```

Property Value

[object](#)

## Xpos

xPos variable setter and getter

```
public int Xpos { get; set; }
```

Property Value

[int](#)

## Ypos

yPos variable setter and getter

```
public int Ypos { get; set; }
```

Property Value

[int](#)

## Methods

### Circle(int, bool)

A method that draws a circle at the current X and Y position with a given radius.

```
public void Circle(int radius, bool filled)
```

Parameters

**radius** [int](#)

Integer which specifies the radius of the circle from the current X and Y position.

**filled** [bool](#)

Boolean which decides whether the circle should be filled inside or hollow.

## Exceptions

CanvasException

CanvasException is thrown when the radius is negative.

## Clear()

Removes any element from the canvas and sets the screen to white.

```
public void Clear()
```

## DrawTo(int, int)

A method that draws a line from the current X and Y position to a different X and Y position specified by the parameters x and y.

```
public void DrawTo(int x, int y)
```

## Parameters

**x** [int](#)

Integer which specifies a location on the horizontal axis.

**y** [int](#)

Integer which specifies a location on the vertical axis.

## Exceptions

CanvasException

CanvasException is thrown when the given x and y values would result in drawing to a position outside the boundary.

## MoveTo(int, int)

A method to move to a (specified by parameters x and y) location, from the current X and Y position.

```
public void MoveTo(int x, int y)
```

### Parameters

x [int](#)

Integer which specifies a location on the horizontal axis.

y [int](#)

Integer which specifies a location on the vertical axis.

### Exceptions

#### CanvasException

CanvasException is thrown when the given x and y values would result in moving to a position outside the boundary.

## Rect(int, int, bool)

Method to draw a rectangle of specific width and height at the current X and Y position.

```
public void Rect(int width, int height, bool filled)
```

### Parameters

width [int](#)

Integer which specifies the width of the rectangle.

height [int](#)

Integer which specifies the height of the rectangle.

**filled** [bool](#)

Boolean which specifies if the drawn rectangle should be filled.

## Exceptions

### CanvasException

CanvasException is thrown when the provided width or height is negative.

## Reset()

Method to move the cursor to the starting position.

```
public void Reset()
```

## Set(int, int)

A method used to initialize drawing components

```
public void Set(int width, int height)
```

## Parameters

**width** [int](#)

Integer which specifies the width of the boundary.

**height** [int](#)

Integer which specifies the height of the boundary.

## Remarks

The method sets a boundary the user can draw on, the boundary is set to the size of pictureBox. The method initializes the graphics object from the bitmap. The method creates a pen using the SetColour class and sets its colour to orange.

## SetColour(Color)

An alternate SetColour method which takes in a Color type parameter to create a pen and a brush.

```
public void SetColour(Color Colour)
```

### Parameters

Colour [Color](#)

Color which specifies a color we want to create a pen with.

### Exceptions

#### CanvasException

CanvasException is thrown when the provided colour is not of the Color type.

## SetColour(int, int, int)

Method to create a pen and a brush based on RGB values.

```
public void SetColour(int red, int green, int blue)
```

### Parameters

red [int](#)

Integer which specifies the redness of the pen and brush.

green [int](#)

Integer which specifies the greenness of the pen and brush.

blue [int](#)

Integer which specifies the blueness of the pen and brush.

### Remarks

The method takes 3 RGB values which are used to create a colour. The colour is then used to create a pen and a brush which the program uses to draw shapes.

## Exceptions

### CanvasException

CanvasException is thrown if any of the given RGB values are smaller than 0 or bigger than 255.

## Tri(int, int)

Method to create a rectangle with a filled triangle inside of it.

```
public void Tri(int width, int height)
```

## Parameters

width [int](#)

Integer which specifies the width of the rectangle.

height [int](#)

Integer which specifies the height of the rectangle.

## Exceptions

### CanvasException

CanvasException is thrown when the provided width or height is negative.

## WriteText(string)

Method to write text on the screen.

```
public void WriteText(string text)
```

## Parameters



text [string](#)

String which is the text to be outputted on the screen.

## getBitmap()

Method to receive the bitmap which contains the drawings.

```
public object getBitmap()
```

Returns

[object](#)

returns the bitmap

Exceptions

CanvasException

CanvasException is thrown if the bitmap is assigned to null.

# Class Form1

Namespace: [ASE Assignment](#)

Assembly: ASE Assignment.dll

Main class of the program. It contains the graphical elements of the interface that the user sees alongside methods that initialize and handle user input

```
public class Form1 : Form, IDropTarget, ISynchronizeInvoke, IWin32Window,
    IBindableComponent, IComponent, IDisposable, IContainerControl
```

## Inheritance

[object](#) ← [MarshalByRefObject](#) ← [Component](#) ← [Control](#) ← [ScrollableControl](#) ← [ContainerControl](#) ← [Form](#) ← Form1

## Implements

[IDropTarget](#), [ISynchronizeInvoke](#), [IWin32Window](#), [IBindableComponent](#), [IComponent](#), [IDisposable](#), [IContainerControl](#)

## Inherited Members

[Form.SetVisibleCore\(bool\)](#), [Form.Activate\(\)](#), [Form.ActivateMdiChild\(Form\)](#), [Form.AddOwnedForm\(Form\)](#), [Form.AdjustFormScrollbars\(bool\)](#), [Form.Close\(\)](#), [Form.CreateAccessibilityInstance\(\)](#), [Form.CreateControlsInstance\(\)](#), [Form.CreateHandle\(\)](#), [Form.DefWndProc\(ref Message\)](#), [Form.ProcessMnemonic\(char\)](#), [Form.CenterToParent\(\)](#), [Form.CenterToScreen\(\)](#), [Form.LayoutMdi\(MdiLayout\)](#), [Form.OnActivated\(EventArgs\)](#), [Form.OnBackgroundImageChanged\(EventArgs\)](#), [Form.OnBackgroundImageLayoutChanged\(EventArgs\)](#), [Form.OnClosing\(CancelEventArgs\)](#), [Form.OnClosed\(EventArgs\)](#), [Form.OnFormClosing\(FormClosingEventArgs\)](#), [Form.OnFormClosed\(FormClosedEventArgs\)](#), [Form.OnCreateControl\(\)](#), [Form.OnDeactivate\(EventArgs\)](#), [Form.OnEnabledChanged\(EventArgs\)](#), [Form.OnEnter\(EventArgs\)](#), [Form.OnFontChanged\(EventArgs\)](#), [Form.OnHandleCreated\(EventArgs\)](#), [Form.OnHandleDestroyed\(EventArgs\)](#), [Form.OnHelpButtonClicked\(CancelEventArgs\)](#), [Form.OnLayout\(LayoutEventArgs\)](#), [Form.OnLoad\(EventArgs\)](#), [Form.OnMaximizedBoundsChanged\(EventArgs\)](#), [Form.OnMaximumSizeChanged\(EventArgs\)](#), [Form.OnMinimumSizeChanged\(EventArgs\)](#), [Form.OnInputLanguageChanged\(InputLanguageChangedEventArgs\)](#), [Form.OnInputLanguageChanging\(InputLanguageChangingEventArgs\)](#), [Form.OnVisibleChanged\(EventArgs\)](#), [Form.OnMdiChildActivate\(EventArgs\)](#), [Form.OnMenuStart\(EventArgs\)](#), [Form.OnMenuComplete\(EventArgs\)](#), [Form.OnPaint\(PaintEventArgs\)](#), [Form.OnResize\(EventArgs\)](#),

[Form.OnDpiChanged\(DpiChangedEventArgs\)](#) , [Form.OnGetDpiScaledSize\(int, int, ref Size\)](#) ,  
[Form.OnRightToLeftLayoutChanged\(EventArgs\)](#) , [Form.OnShown\(EventArgs\)](#) ,  
[Form.OnTextChanged\(EventArgs\)](#) , [Form.ProcessCmdKey\(ref Message, Keys\)](#) ,  
[Form.ProcessDialogKey\(Keys\)](#) , [Form.ProcessDialogChar\(char\)](#) ,  
[Form.ProcessKeyPreview\(ref Message\)](#) , [Form.ProcessTabKey\(bool\)](#) ,  
[Form.RemoveOwnedForm\(Form\)](#) , [Form.Select\(bool, bool\)](#) ,  
[Form.GetScaledBounds\(Rectangle, SizeF, BoundsSpecified\)](#) ,  
[Form.ScaleControl\(SizeF, BoundsSpecified\)](#) , [Form.SetBoundsCore\(int, int, int, int, BoundsSpecified\)](#) ,  
[Form.SetClientSizeCore\(int, int\)](#) , [Form.SetDesktopBounds\(int, int, int, int\)](#) ,  
[Form.SetDesktopLocation\(int, int\)](#) , [Form.Show\(IWin32Window\)](#) , [Form.ShowDialog\(\)](#) ,  
[Form.ShowDialog\(IWin32Window\)](#) , [Form.ToString\(\)](#) , [Form.UpdateDefaultButton\(\)](#) ,  
[Form.OnResizeBegin\(EventArgs\)](#) , [Form.OnResizeEnd\(EventArgs\)](#) ,  
[Form.OnStyleChanged\(EventArgs\)](#) , [Form.ValidateChildren\(\)](#) ,  
[Form.ValidateChildren\(ValidationConstraints\)](#) , [Form.WndProc\(ref Message\)](#) , [Form.AcceptButton](#) ,  
[Form.ActiveForm](#) , [Form.ActiveMdiChild](#) , [Form.AllowTransparency](#) , [Form.AutoScroll](#) ,  
[Form.AutoSize](#) , [Form.AutoSizeMode](#) , [Form.AutoValidate](#) , [Form.BackColor](#) ,  
[Form.FormBorderStyle](#) , [Form.CancelButton](#) , [Form.ClientSize](#) , [Form.ControlBox](#) ,  
[Form.CreateParams](#) , [Form.DefaultImeMode](#) , [Form.DefaultSize](#) , [Form.DesktopBounds](#) ,  
[Form.DesktopLocation](#) , [Form.DialogResult](#) , [Form.HelpButton](#) , [Form.Icon](#) , [Form.IsMdiChild](#) ,  
[Form.IsMdiContainer](#) , [Form.IsRestrictedWindow](#) , [Form.KeyPreview](#) , [Form.Location](#) ,  
[Form.MaximizedBounds](#) , [Form.MaximumSize](#) , [Form.MainMenuStrip](#) , [Form.MinimumSize](#) ,  
[Form.MaximizeBox](#) , [Form.MdiChildren](#) , [Form.MdiChildrenMinimizedAnchorBottom](#) ,  
[Form.MdiParent](#) , [Form.MinimizeBox](#) , [Form.Modal](#) , [Form.Opacity](#) , [Form.OwnedForms](#) ,  
[Form.Owner](#) , [Form.RestoreBounds](#) , [Form.RightToLeftLayout](#) , [Form.ShowInTaskbar](#) ,  
[Form.ShowIcon](#) , [Form.ShowWithoutActivation](#) , [Form.Size](#) , [Form.SizeGripStyle](#) ,  
[Form.StartPosition](#) , [Form.Text](#) , [Form.TopLevel](#) , [Form.TopMost](#) , [Form.TransparencyKey](#) ,  
[Form.WindowState](#) , [Form.AutoSizeChanged](#) , [Form.AutoValidateChanged](#) ,  
[Form.HelpButtonClicked](#) , [Form.MaximizedBoundsChanged](#) , [Form.MaximumSizeChanged](#) ,  
[Form.MinimumSizeChanged](#) , [Form.Activated](#) , [Form.Deactivate](#) , [Form.FormClosing](#) ,  
[Form.FormClosed](#) , [Form.Load](#) , [Form.MdiChildActivate](#) , [Form.MenuComplete](#) ,  
[Form.MenuStart](#) , [Form.InputLanguageChanged](#) , [Form.InputLanguageChanging](#) ,  
[Form.RightToLeftLayoutChanged](#) , [Form.Shown](#) , [Form.DpiChanged](#) , [Form.ResizeBegin](#) ,  
[Form.ResizeEnd](#) , [ContainerControl.OnAutoValidateChanged\(EventArgs\)](#) ,  
[ContainerControl.OnParentChanged\(EventArgs\)](#) , [ContainerControl.PerformAutoScale\(\)](#) ,  
[ContainerControl.RescaleConstantsForDpi\(int, int\)](#) , [ContainerControl.Validate\(\)](#) ,  
[ContainerControl.Validate\(bool\)](#) , [ContainerControl.AutoScaleDimensions](#) ,  
[ContainerControl.AutoScaleFactor](#) , [ContainerControl.AutoScaleMode](#) ,  
[ContainerControl.BindingContext](#) , [ContainerControl.CanEnableIme](#) ,  
[ContainerControl.ActiveControl](#) , [ContainerControl.CurrentAutoScaleDimensions](#) ,  
[ContainerControl.ParentForm](#) , [ScrollableControl.ScrollStateAutoScrolling](#) ,

[ScrollableControl.ScrollStateHScrollVisible](#) , [ScrollableControl.ScrollStateVScrollVisible](#) ,  
[ScrollableControl.ScrollStateUserHasScrolled](#) , [ScrollableControl.ScrollStateFullDrag](#) ,  
[ScrollableControl.GetScrollState\(int\)](#) , [ScrollableControl.OnMouseWheel\(MouseEventArgs\)](#) ,  
[ScrollableControl.OnRightToLeftChanged\(EventArgs\)](#) ,  
[ScrollableControl.OnPaintBackground\(PaintEventArgs\)](#) ,  
[ScrollableControl.OnPaddingChanged\(EventArgs\)](#) , [ScrollableControl.SetDisplayRectLocation\(int, int\)](#) ,  
[ScrollableControl.ScrollControlIntoView\(Control\)](#) , [ScrollableControl.ScrollToControl\(Control\)](#) ,  
[ScrollableControl.OnScroll\(ScrollEventArgs\)](#) , [ScrollableControl.SetAutoScrollMargin\(int, int\)](#) ,  
[ScrollableControl.SetScrollState\(int, bool\)](#) , [ScrollableControl.AutoScrollMargin](#) ,  
[ScrollableControl.AutoScrollPosition](#) , [ScrollableControl.AutoScrollMinSize](#) ,  
[ScrollableControl.DisplayRectangle](#) , [ScrollableControl.HScroll](#) , [ScrollableControl.HorizontalScroll](#) ,  
[ScrollableControl.VScroll](#) , [ScrollableControl.VerticalScroll](#) , [ScrollableControl.Scroll](#) ,  
[Control.GetAccessibilityObjectById\(int\)](#) , [Control.SetAutoSizeMode\(AutoSizeMode\)](#) ,  
[Control.GetAutoSizeMode\(\)](#) , [Control.GetPreferredSize\(Size\)](#) ,  
[Control.AccessibilityNotifyClients\(AccessibleEvents, int\)](#) ,  
[Control.AccessibilityNotifyClients\(AccessibleEvents, int, int\)](#) , [Control.BeginInvoke\(Delegate\)](#) ,  
[Control.BeginInvoke\(Action\)](#) , [Control.BeginInvoke\(Delegate, params object\[\]\)](#) ,  
[Control.BringToFront\(\)](#) , [Control.Contains\(Control\)](#) , [Control.CreateGraphics\(\)](#) ,  
[Control.CreateControl\(\)](#) , [Control.DestroyHandle\(\)](#) , [Control.DoDragDrop\(object, DragDropEffects\)](#) ,  
[Control.DrawToBitmap\(Bitmap, Rectangle\)](#) , [Control.EndInvoke\(IAsyncResult\)](#) , [Control.FindForm\(\)](#) ,  
[Control.GetTopLevel\(\)](#) , [Control.RaiseKeyEvent\(object, KeyEventArgs\)](#) ,  
[Control.RaiseMouseEvent\(object, MouseEventArgs\)](#) , [Control.Focus\(\)](#) ,  
[Control.FromChildHandle\(IntPtr\)](#) , [Control.FromHandle\(IntPtr\)](#) ,  
[Control.GetChildAtPoint\(Point, GetChildAtPointSkip\)](#) , [Control.GetChildAtPoint\(Point\)](#) ,  
[Control.GetContainerControl\(\)](#) , [Control.GetNextControl\(Control, bool\)](#) ,  
[Control.GetStyle\(ControlStyles\)](#) , [Control.Hide\(\)](#) , [Control.InitLayout\(\)](#) , [Control.Invalidate\(Region\)](#) ,  
[Control.Invalidate\(Region, bool\)](#) , [Control.Invalidate\(\)](#) , [Control.Invalidate\(bool\)](#) ,  
[Control.Invalidate\(Rectangle\)](#) , [Control.Invalidate\(Rectangle, bool\)](#) , [Control.Invoke\(Action\)](#) ,  
[Control.Invoke\(Delegate\)](#) , [Control.Invoke\(Delegate, params object\[\]\)](#) ,  
[Control.Invoke<T>\(Func<T>\)](#) , [Control.InvokePaint\(Control, PaintEventArgs\)](#) ,  
[Control.InvokePaintBackground\(Control, PaintEventArgs\)](#) , [Control.IsKeyLocked\(Keys\)](#) ,  
[Control.IsInputChar\(char\)](#) , [Control.IsInputKey\(Keys\)](#) , [Control.IsMnemonic\(char, string\)](#) ,  
[Control.LogicalToDeviceUnits\(int\)](#) , [Control.LogicalToDeviceUnits\(Size\)](#) ,  
[Control.ScaleBitmapLogicalToDevice\(ref Bitmap\)](#) , [Control.NotifyInvalidate\(Rectangle\)](#) ,  
[Control.InvokeOnClick\(Control, EventArgs\)](#) , [Control.OnAutoSizeChanged\(EventArgs\)](#) ,  
[Control.OnBackColorChanged\(EventArgs\)](#) , [Control.OnBindingContextChanged\(EventArgs\)](#) ,  
[Control.OnCausesValidationChanged\(EventArgs\)](#) , [Control.OnContextMenuStripChanged\(EventArgs\)](#) ,  
[Control.OnCursorChanged\(EventArgs\)](#) , [Control.OnDockChanged\(EventArgs\)](#) ,  
[Control.OnForeColorChanged\(EventArgs\)](#) , [Control.OnNotifyMessage\(Message\)](#) ,  
[Control.OnParentBackColorChanged\(EventArgs\)](#) ,

[Control.OnParentBackgroundImageChanged\(EventArgs\)](#),  
[Control.OnParentBindingContextChanged\(EventArgs\)](#), [Control.OnParentCursorChanged\(EventArgs\)](#),  
[Control.OnParentEnabledChanged\(EventArgs\)](#), [Control.OnParentFontChanged\(EventArgs\)](#),  
[Control.OnParentForeColorChanged\(EventArgs\)](#), [Control.OnParentRightToLeftChanged\(EventArgs\)](#),  
[Control.OnParentVisibleChanged\(EventArgs\)](#), [Control.OnPrint\(PaintEventArgs\)](#),  
[Control.OnTabIndexChanged\(EventArgs\)](#), [Control.OnTabStopChanged\(EventArgs\)](#),  
[Control.OnClick\(EventArgs\)](#), [Control.OnClientSizeChanged\(EventArgs\)](#),  
[Control.OnControlAdded\(ControlEventArgs\)](#), [Control.OnControlRemoved\(ControlEventArgs\)](#),  
[Control.OnLocationChanged\(EventArgs\)](#), [Control.OnDoubleClick\(EventArgs\)](#),  
[Control.OnDragEnter\(DragEventArgs\)](#), [Control.OnDragOver\(DragEventArgs\)](#),  
[Control.OnDragLeave\(EventArgs\)](#), [Control.OnDragDrop\(DragEventArgs\)](#),  
[Control.OnGiveFeedback\(GiveFeedbackEventArgs\)](#), [Control.InvokeGotFocus\(Control, EventArgs\)](#),  
[Control.OnGotFocus\(EventArgs\)](#), [Control.OnHelpRequested\(HelpEventArgs\)](#),  
[Control.OnInvalidated\(InvalidateEventArgs\)](#), [Control.OnKeyDown\(KeyEventArgs\)](#),  
[Control.OnKeyPress\(KeyPressEventArgs\)](#), [Control.OnKeyUp\(KeyEventArgs\)](#),  
[Control.OnLeave\(EventArgs\)](#), [Control.InvokeLostFocus\(Control, EventArgs\)](#),  
[Control.OnLostFocus\(EventArgs\)](#), [Control.OnMarginChanged\(EventArgs\)](#),  
[Control.OnMouseDoubleClick\(MouseEventArgs\)](#), [Control.OnMouseClick\(MouseEventArgs\)](#),  
[Control.OnMouseCaptureChanged\(EventArgs\)](#), [Control.OnMouseDown\(MouseEventArgs\)](#),  
[Control.OnMouseEnter\(EventArgs\)](#), [Control.OnMouseLeave\(EventArgs\)](#),  
[Control.OnDpiChangedBeforeParent\(EventArgs\)](#), [Control.OnDpiChangedAfterParent\(EventArgs\)](#),  
[Control.OnMouseHover\(EventArgs\)](#), [Control.OnMouseMove\(MouseEventArgs\)](#),  
[Control.OnMouseUp\(MouseEventArgs\)](#), [Control.OnMove\(EventArgs\)](#),  
[Control.OnQueryContinueDrag\(QueryContinueDragEventArgs\)](#),  
[Control.OnRegionChanged\(EventArgs\)](#), [Control.OnPreviewKeyDown\(PreviewKeyDownEventArgs\)](#),  
[Control.OnSizeChanged\(EventArgs\)](#), [Control.OnChangeUICues\(UICuesEventArgs\)](#),  
[Control.OnSystemColorsChanged\(EventArgs\)](#), [Control.OnValidating\(CancelEventArgs\)](#),  
[Control.OnValidated\(EventArgs\)](#), [Control.PerformLayout\(\)](#), [Control.PerformLayout\(Control, string\)](#),  
[Control.PointToClient\(Point\)](#), [Control.PointToScreen\(Point\)](#),  
[Control.PreProcessMessage\(ref Message\)](#), [Control.PreProcessControlMessage\(ref Message\)](#),  
[Control.ProcessKeyEventArgs\(ref Message\)](#), [Control.ProcessKeyMessage\(ref Message\)](#),  
[Control.RaiseDragEvent\(object, DragEventArgs\)](#), [Control.RaisePaintEvent\(object, PaintEventArgs\)](#),  
[Control.RecreateHandle\(\)](#), [Control.RectangleToClient\(Rectangle\)](#),  
[Control.RectangleToScreen\(Rectangle\)](#), [Control.ReflectMessage\(IntPtr, ref Message\)](#),  
[Control.Refresh\(\)](#), [Control.ResetMouseEventArgs\(\)](#), [Control.ResetText\(\)](#), [Control.ResumeLayout\(\)](#),  
[Control.ResumeLayout\(bool\)](#), [Control.Scale\(SizeF\)](#), [Control.Select\(\)](#),  
[Control.SelectNextControl\(Control, bool, bool, bool, bool\)](#), [Control.SendToBack\(\)](#),  
[Control.SetBounds\(int, int, int, int\)](#), [Control.SetBounds\(int, int, int, int, BoundsSpecified\)](#),  
[Control.SizeFromClientSize\(Size\)](#), [Control.SetStyle\(ControlStyles, bool\)](#), [Control.SetTopLevel\(bool\)](#),  
[Control.RtlTranslateAlignment\(HorizontalAlignment\)](#),

[Control.RtlTranslateAlignment\(LeftRightAlignment\)](#),  
[Control.RtlTranslateAlignment\(ContentAlignment\)](#),  
[Control.RtlTranslateHorizontal\(HorizontalAlignment\)](#),  
[Control.RtlTranslateLeftRight\(LeftRightAlignment\)](#), [Control.RtlTranslateContent\(ContentAlignment\)](#),  
[Control.Show\(\)](#), [Control.SuspendLayout\(\)](#), [Control.Update\(\)](#), [Control.UpdateBounds\(\)](#),  
[Control.UpdateBounds\(int, int, int, int\)](#), [Control.UpdateBounds\(int, int, int, int, int, int\)](#),  
[Control.UpdateZOrder\(\)](#), [Control.UpdateStyles\(\)](#), [Control.OnImeModeChanged\(EventArgs\)](#),  
[Control.AccessibilityObject](#), [Control.AccessibleDefaultActionDescription](#),  
[Control.AccessibleDescription](#), [Control.AccessibleName](#), [Control.AccessibleRole](#),  
[Control.AllowDrop](#), [Control.Anchor](#), [Control.AutoScrollOffset](#), [Control.LayoutEngine](#),  
[Control.BackgroundImage](#), [Control.BackgroundImageLayout](#), [Control.Bottom](#), [Control.Bounds](#),  
[Control.CanFocus](#), [Control.CanRaiseEvents](#), [Control.CanSelect](#), [Control.Capture](#),  
[Control.CausesValidation](#), [Control.CheckForIllegalCrossThreadCalls](#), [Control.ClientRectangle](#),  
[Control.CompanyName](#), [Control.ContainsFocus](#), [Control.ContextMenuStrip](#), [Control.Controls](#),  
[Control.Created](#), [Control.Cursor](#), [Control.DataBindings](#), [Control.DefaultBackColor](#),  
[Control.DefaultCursor](#), [Control.DefaultFont](#), [Control.DefaultForeColor](#), [Control.DefaultMargin](#),  
[Control.DefaultMaximumSize](#), [Control.DefaultMinimumSize](#), [Control.DefaultPadding](#),  
[Control.DeviceDpi](#), [Control.IsDisposed](#), [Control.Disposing](#), [Control.Dock](#),  
[Control.DoubleBuffered](#), [Control.Enabled](#), [Control.Focused](#), [Control.Font](#),  
[Control.FontHeight](#), [Control.ForeColor](#), [Control.Handle](#), [Control.HasChildren](#), [Control.Height](#),  
[Control.IsHandleCreated](#), [Control.InvokeRequired](#), [Control.IsAccessible](#),  
[Control.IsAncestorSiteInDesignMode](#), [Control.IsMirrored](#), [Control.Left](#), [Control.Margin](#),  
[Control.ModifierKeys](#), [Control.MouseButtons](#), [Control.MousePosition](#), [Control.Name](#),  
[Control.Parent](#), [Control.ProductName](#), [Control.ProductVersion](#), [Control.RecreatingHandle](#),  
[Control.Region](#), [Control.RenderRightToLeft](#), [Control.ResizeRedraw](#), [Control.Right](#),  
[Control.RightToLeft](#), [Control.ScaleChildren](#), [Control.Site](#), [Control.TabIndex](#), [Control.TabStop](#),  
[Control.Tag](#), [Control.Top](#), [Control.TopLevelControl](#), [Control.ShowKeyboardCues](#),  
[Control.ShowFocusCues](#), [Control.UseWaitCursor](#), [Control.Visible](#), [Control.Width](#),  
[Control.PreferredSize](#), [Control.Padding](#), [Control.ImeMode](#), [Control.ImeModeBase](#),  
[Control.PropagatingImeMode](#), [Control.BackColorChanged](#), [Control.BackgroundImageChanged](#),  
[Control.BackgroundImageLayoutChanged](#), [Control.BindingContextChanged](#),  
[Control.CausesValidationChanged](#), [Control.ClientSizeChanged](#),  
[Control.ContextMenuStripChanged](#), [Control.CursorChanged](#), [Control.DockChanged](#),  
[Control.EnabledChanged](#), [Control.FontChanged](#), [Control.ForeColorChanged](#),  
[Control.LocationChanged](#), [Control.MarginChanged](#), [Control.RegionChanged](#),  
[Control.RightToLeftChanged](#), [Control.SizeChanged](#), [Control.TabIndexChanged](#),  
[Control.TabStopChanged](#), [Control.TextChanged](#), [Control.VisibleChanged](#), [Control.Click](#),  
[Control.ControlAdded](#), [Control.ControlRemoved](#), [Control.DragDrop](#), [Control.DragEnter](#),  
[Control.DragOver](#), [Control.DragLeave](#), [Control.GiveFeedback](#), [Control.HandleCreated](#),  
[Control.HandleDestroyed](#), [Control.HelpRequested](#), [Control.Invalidated](#),

[Control.PaddingChanged](#) , [Control.Paint](#) , [Control.QueryContinueDrag](#) ,  
[Control.QueryAccessibilityHelp](#) , [Control.DoubleClick](#) , [Control.Enter](#) , [Control.GotFocus](#) ,  
[Control.KeyDown](#) , [Control.KeyPress](#) , [Control.KeyUp](#) , [Control.Layout](#) , [Control.Leave](#) ,  
[Control.LostFocus](#) , [Control.MouseClick](#) , [Control.MouseDoubleClick](#) ,  
[Control.MouseCaptureChanged](#) , [Control.MouseDown](#) , [Control.MouseEnter](#) ,  
[Control.MouseLeave](#) , [Control.DpiChangedBeforeParent](#) , [Control.DpiChangedAfterParent](#) ,  
[Control.MouseHover](#) , [Control.MouseMove](#) , [Control.MouseUp](#) , [Control.MouseWheel](#) ,  
[Control.Move](#) , [Control.PreviewKeyDown](#) , [Control.Resize](#) , [Control.ChangeUICues](#) ,  
[Control.StyleChanged](#) , [Control.SystemColorsChanged](#) , [Control.Validating](#) , [Control.Validated](#) ,  
[Control.ParentChanged](#) , [Control.ImeModeChanged](#) , [Component.Dispose\(\)](#) ,  
[Component.GetService\(Type\)](#) , [Component.Container](#) , [Component.DesignMode](#) ,  
[Component.Events](#) , [Component.Disposed](#) , [MarshalByRefObject.GetLifetimeService\(\)](#) ,  
[MarshalByRefObject.InitializeLifetimeService\(\)](#) , [MarshalByRefObject.MemberwiseClone\(bool\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Form1()

Constructor for the Form1 class, its used to initialize all elements necessary to to view, draw and take input from the user

```
public Form1()
```

## Methods

### Dispose(bool)

Clean up any resources being used.

```
protected override void Dispose(bool disposing)
```

### Parameters

**disposing** [bool](#)

true if managed resources should be disposed; otherwise, false.

# Class ownCommandfactory


Namespace: [ASE Assignment](#)

Assembly: ASE Assignment.dll

The class extends CommandFactory from BOOSE. Both classes receive commands from the user as input and create new objects of their corresponding classes, if they exist in the factory.

```
public class ownCommandfactory : CommandFactory, ICommandFactory
```








## Inheritance

[object](#)  ← CommandFactory ← ownCommandfactory

## Implements

ICommandFactory

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Methods

### MakeCommand(string)

Method to see if a command exists, if it does; it creates a new object of the corresponding command.

```
public override ICommand MakeCommand(string commandType)
```

## Parameters

commandType [string](#) 

String which is the command passed in by the Parser after user inputs their text.

## Returns

ICommand



# Class ownPenColour


Namespace: [ASE Assignment](#)

Assembly: ASE Assignment.dll

Class used to create a pen of a specified colour

```
public class ownPenColour : CommandThreeParameters, ICommand
```










## Inheritance

[object](#)  ← Command ← CanvasCommand ← CommandOneParameter ← CommandTwoParameters ← CommandThreeParameters ← ownPenColour

## Implements

ICommand

## Inherited Members

CommandThreeParameters.param3 , CommandThreeParameters.param3unprocessed , [CommandThreeParameters.CheckParameters\(string\[\]\)](#)  , CommandTwoParameters.param2 , CommandTwoParameters.param2unprocessed , CommandOneParameter.param1 , CommandOneParameter.param1unprocessed , CanvasCommand.yPos , CanvasCommand.xPos , CanvasCommand.canvas , CanvasCommand.Canvas , Command.program , Command.parameterList , Command.parameters , Command.paramsint , [Command.Set\(StoredProgram, string\)](#)  , Command.Compile() , [Command.ProcessParameters\(string\)](#)  , Command.ToString() , Command.Program , Command.Name , Command.ParameterList , Command.Parameters , Command.Paramsint , [object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#) 

## Methods

### Execute()

Method used to create a pen with a specific RGB colour by calling the Super class version of the Execute() command, alongside calling the SetColour by using the Canvas getter which will return the Draws class.

```
public override void Execute()
```

# Namespace UnitTestsASE

## Classes

### [CircleTests](#)

Test Class used to test the Circle command.

### [ColourTests](#)

Test class used to test the setColour methods.

### [DrawtoTests](#)

Test Class used to test the DrawTo command

### [MoveToTests](#)

Test Class used to test the MoveTo command.

### [MultilineTest](#)

Test class to test the multiline functionality of the program

### [RectangleTests](#)

Test class used to test the Rect method.

### [ResetTest](#)

Test class used to test the reset method.

### [TriangleTests](#)

Test class to test the Triangle method

# Class CircleTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test Class used to test the Circle command.

```
[TestClass]
public class CircleTests
```

## Inheritance

[object](#) ← CircleTests

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Methods

## InvalidRadiusTestThrowsCanvasException()

Test method used to see if the Circle method will throw a CanvasException when a negative radius is passed.

```
[TestMethod]
public void InvalidRadiusTestThrowsCanvasException()
```

## XPositionAfterCircleCommand()

Test method used to see if the X position didn't change after calling the Circle method.

```
[TestMethod]
public void XPositionAfterCircleCommand()
```

## XPositionAfterDrawToThenCircle()

Test method used to see if the X position set by the DrawTo command didn't change because of the Circle command

```
[TestMethod]  
public void XPositionAfterDrawToThenCircle()
```

## XPositionAfterMoveToThenCircle()

Test method used to see if the X position set by the MoveTo command didn't change because of the Circle command

```
[TestMethod]  
public void XPositionAfterMoveToThenCircle()
```

## YPositionAfterCircleCommand()

Test method used to see if the Y position didn't change after calling the Circle method.

```
[TestMethod]  
public void YPositionAfterCircleCommand()
```

## YPositionAfterDrawToThenCircle()

Test method used to see if the X position set by the DrawTo command didn't change because of the Circle command

```
[TestMethod]  
public void YPositionAfterDrawToThenCircle()
```

## YPositionAfterMoveToThenCircle()

Test method used to see if the Y position set by the MoveTo command didn't change because of the Circle command

```
[TestMethod]  
public void YPositionAfterMoveToThenCircle()
```

# Class ColourTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll








Test class used to test the setColour methods.

```
[TestClass]
public class ColourTests
```

## Inheritance

[object](#)  ← ColourTests

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

# Methods

## ColorTypeColorSetTest()

Test method used to see if the alternate setColour method correctly creates a pen with a specified colour of Color type

```
[TestMethod]
public void ColorTypeColorSetTest()
```

## RGBColorSetTest()

Test method used to see if the setColour method correctly creates a pen with the specified RGB colour

```
[TestMethod]
public void RGBColorSetTest()
```

## RGBInvalidBlueColourSetShouldThrowCanvasException()

Test method to see if blue negative and values exceeding 255 throw a CanvasException

```
[TestMethod]  
public void RGBInvalidBlueColourSetShouldThrowCanvasException()
```

## RGBInvalidGreenColourSetShouldThrowCanvasException()

Test method to see if green negative and values exceeding 255 throw a CanvasException

```
[TestMethod]  
public void RGBInvalidGreenColourSetShouldThrowCanvasException()
```

## RGBInvalidRedColourSetShouldThrowCanvasException()

Test method to see if red negative and values exceeding 255 throw a CanvasException

```
[TestMethod]  
public void RGBInvalidRedColourSetShouldThrowCanvasException()
```

# Class DrawtoTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test Class used to test the DrawTo command

```
[TestClass]
public class DrawtoTests
```

## Inheritance

[object](#) ← DrawtoTests

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Methods

## DrawToXTestCorrect()

Test method used to see if the X position of the cursor correctly changes to a specified expected value after the DrawTo command is called, with the x parameter being set to the specified expected value.

```
[TestMethod]
public void DrawToXTestCorrect()
```

## DrawToXTestIncorrect()

Test method used to see if a random expected value causes the test to pass when moving the cursor using the DrawTo command

```
[TestMethod]
public void DrawToXTestIncorrect()
```



## DrawToYTestCorrect()

Test method used to see if the Y position of the cursor correctly changes to a specified expected value after the DrawTo command is called, with the Y parameter being set to the specified expected value.

```
[TestMethod]  
public void DrawToYTestCorrect()
```

## DrawToYTestIncorrect()

Test method used to see if a random expected value causes the test to pass when moving the cursor using the DrawTo command

```
[TestMethod]  
public void DrawToYTestIncorrect()
```

## NegativeOutOfBoundsXshouldThrowCanvasException()

Test method used to see if calling the DrawTo method with X parameter values being negative throws a CanvasException

```
[TestMethod]  
public void NegativeOutOfBoundsXshouldThrowCanvasException()
```

## NegativeOutOfBoundsYshouldThrowCanvasException()

Test method used to see if calling the DrawTo method with Y parameter values being negative throws a CanvasException

```
[TestMethod]  
public void NegativeOutOfBoundsYshouldThrowCanvasException()
```

## PositiveOutOfBoundsYshouldThrowCanvasException()

Test method used to see if calling the DrawTo method with Y parameter values exceeding the canvas boundary throws a CanvasException

```
[TestMethod]  
public void PositiveOutOfBoundsYshouldThrowCanvasException()
```

## positiveOutOfBoundsXshouldThrowCanvasException()

Test method used to see if calling the DrawTo method with X parameter values exceeding the canvas boundary throws a CanvasException

```
[TestMethod]  
public void positiveOutOfBoundsXshouldThrowCanvasException()
```

# Class MoveToTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test Class used to test the MoveTo command.

```
[TestClass]
public class MoveToTests
```

## Inheritance

[object](#) ← MoveToTests

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Methods

## MoveToXTestCorrect()

Test method used to see if the X position of the cursor correctly changes to a specified expected value after the MoveTo command is called, with the x parameter being set to the specified expected value.

```
[TestMethod]
public void MoveToXTestCorrect()
```

## MoveToXTestIncorrect()

Test method used to see if a random expected value causes the test to pass when moving the cursor using the MoveTo command

```
[TestMethod]
public void MoveToXTestIncorrect()
```

## MoveToYTestCorrect()

Test method used to see if the Y position of the cursor correctly changes to a specified expected value after the MoveTo command is called, with the Y parameter being set to the specified expected value.

```
[TestMethod]  
public void MoveToYTestCorrect()
```

## MoveToYTestIncorrect()

Test method used to see if a random expected value causes the test to pass when moving the cursor using the MoveTo command

```
[TestMethod]  
public void MoveToYTestIncorrect()
```

## NegativeOutOfBoundsXshouldThrowCanvasException()

Test method used to see if calling the MoveTo method with X parameter values being negative throws a CanvasException

```
[TestMethod]  
public void NegativeOutOfBoundsXshouldThrowCanvasException()
```

## NegativeOutOfBoundsYshouldThrowCanvasException()

Test method used to see if calling the MoveTo method with Y parameter values being negative throws a CanvasException

```
[TestMethod]  
public void NegativeOutOfBoundsYshouldThrowCanvasException()
```

## PositiveOutOfBoundsYshouldThrowCanvasException()

Test method used to see if calling the MoveTo method with Y parameter values exceeding the canvas boundary throws a CanvasException

```
[TestMethod]  
public void PositiveOutOfBoundsYshouldThrowCanvasException()
```

## positiveOutOfBoundsXshouldThrowCanvasException()

Test method used to see if calling the MoveTo method with X parameter values exceeding the canvas boundary throws a CanvasException

```
[TestMethod]  
public void positiveOutOfBoundsXshouldThrowCanvasException()
```

# Class MultilineTest

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test class to test the multiline functionality of the program

```
[TestClass]
public class MultilineTest
```

## Inheritance

[object](#) ← MultilineTest

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Methods

### XMultilineCheckUsingCircleMoveToRectTriangle()

Test method to see if the X position set by the MoveTo command isn't changed after calling the rectangle and triangle methods.

```
[TestMethod]
public void XMultilineCheckUsingCircleMoveToRectTriangle()
```

### XMultilineCheckUsingCircleRectTriangle()

Test method to see if the X position remains the same after a circle, rectangle and triangle method calls.

```
[TestMethod]
public void XMultilineCheckUsingCircleRectTriangle()
```

### XMultilineCheckUsingDrawTo()

Test class to see if the correct X value is set after multiple DrawTo method calls.

```
[TestMethod]  
public void XMultilineCheckUsingDrawTo()
```

## XMultilineCheckUsingDrawtoCircleRectTriangle()

Test method to see if the X position set by the DrawTo command isn't changed after calling the circle, rectangle and triangle methods.

```
[TestMethod]  
public void XMultilineCheckUsingDrawtoCircleRectTriangle()
```

## XMultilineCheckUsingMoveTo()

Test class to see if the correct X value is set after multiple MoveTo method calls.

```
[TestMethod]  
public void XMultilineCheckUsingMoveTo()
```

## XMultilineCheckUsingMoveToCircleRectTriangle()

Test method to see if the X position set by the MoveTo command isn't changed after calling the circle, rectangle and triangle methods.

```
[TestMethod]  
public void XMultilineCheckUsingMoveToCircleRectTriangle()
```

## YMultilineCheckUsingCircleMoveToRectTriangle()

Test method to see if the Y position set by the MoveTo command isn't changed after calling the rectangle and triangle methods.

```
[TestMethod]
```

```
public void YMultilineCheckUsingCircleMoveToRectTriangle()
```

## YMultilineCheckUsingCircleRectTriangle()

Test method to see if the Y position remains the same after a circle, rectangle and triangle method calls.

```
[TestMethod]  
public void YMultilineCheckUsingCircleRectTriangle()
```

## YMultilineCheckUsingDrawTo()

Test class to see if the correct Y value is set after multiple DrawTo method calls.

```
[TestMethod]  
public void YMultilineCheckUsingDrawTo()
```

## YMultilineCheckUsingDrawtoCircleRectTriangle()

Test method to see if the X position set by the DrawTo command isn't changed after calling the circle, rectangle and triangle methods.

```
[TestMethod]  
public void YMultilineCheckUsingDrawtoCircleRectTriangle()
```

## YMultilineCheckUsingMoveTo()

Test class to see if the correct Y value is set after multiple MoveTo method calls.

```
[TestMethod]  
public void YMultilineCheckUsingMoveTo()
```

## YMultilineCheckUsingMoveToCircleRectTriangle()



Test method to see if the Y position set by the MoveTo command isn't changed after calling the circle, rectangle and triangle methods.

```
[TestMethod]  
public void YMultilineCheckUsingMoveToCircleRectTriangle()
```

# Class RectangleTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test class used to test the Rect method.

```
[TestClass]
public class RectangleTests
```

## Inheritance

[object](#) ← RectangleTests

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Methods

### RectangleThrowsExceptionWhenNegativeHeightIsPassed()

Test method used to see if the Rect method throws a CanvasException when a negative Height is passed

```
[TestMethod]
public void RectangleThrowsExceptionWhenNegativeHeightIsPassed()
```

### RectangleThrowsExceptionWhenNegativeWidthIsPassed()

Test method used to see if the Rect method throws a CanvasException when a negative Width is passed

```
[TestMethod]
public void RectangleThrowsExceptionWhenNegativeWidthIsPassed()
```

### XPositionAfterDrawToThenRectangleCommand()

Test method used to see if the X position set by the DrawTo command didn't change because of the Rect command

```
[TestMethod]  
public void XPositionAfterDrawToThenRectangleCommand()
```

## XPositionAfterMoveToThenRectangleCommand()

Test method used to see if the X position set by the MoveTo command didn't change because of the Rect command

```
[TestMethod]  
public void XPositionAfterMoveToThenRectangleCommand()
```

## XPositionAfterRectangleCommand()

Test method used to see if the X position didn't change after calling the Rect method.

```
[TestMethod]  
public void XPositionAfterRectangleCommand()
```

## YPositionAfterDrawToThenRectangleCommand()

Test method used to see if the Y position set by the DrawTo command didn't change because of the Rect command

```
[TestMethod]  
public void YPositionAfterDrawToThenRectangleCommand()
```

## YPositionAfterMoveToThenRectangleCommand()

Test method used to see if the Y position set by the MoveTo command didn't change because of the Rect command

```
[TestMethod]  
public void YPositionAfterMoveToThenRectangleCommand()
```

## YPositionAfterRectangleCommand()

Test method used to see if the Y position didn't change after calling the Rect method.

```
[TestMethod]  
public void YPositionAfterRectangleCommand()
```

# Class ResetTest

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test class used to test the reset method.

```
[TestClass]  
public class ResetTest
```

## Inheritance

[object](#) ← ResetTest

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Methods

## XResetTestAfterMoving()

Test method used to see if the X position correctly changed from the reset command after a movement command was called

```
[TestMethod]  
public void XResetTestAfterMoving()
```

## XResetTestBeforeMoving()

Test method used to see if the X position didn't change from the reset command before any movement command was called

```
[TestMethod]  
public void XResetTestBeforeMoving()
```

## YResetTestAfterMoving()

Test method used to see if the Y position correctly changed from the reset command after a movement command was called

```
[TestMethod]  
public void YResetTestAfterMoving()
```

## YResetTestBeforeMoving()

Test method used to see if the Y position didn't change from the reset command before any movement command was called

```
[TestMethod]  
public void YResetTestBeforeMoving()
```

# Class TriangleTests

Namespace: [UnitTestsASE](#)

Assembly: UnitTestsASE.dll

Test class to test the Triangle method

```
[TestClass]
public class TriangleTests
```

## Inheritance

[object](#) ← TriangleTests

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Methods

### TriangleThrowsExceptionWhenNegativeHeightIsPassed()

Test method to see if the Tri method throws a CanvasException when negative height is passed

```
[TestMethod]
public void TriangleThrowsExceptionWhenNegativeHeightIsPassed()
```

### TriangleThrowsExceptionWhenNegativeWidthIsPassed()

Test method to see if the Tri method throws a CanvasException when negative width is passed

```
[TestMethod]
public void TriangleThrowsExceptionWhenNegativeWidthIsPassed()
```

### XPositionAfterDrawToThenTriangleCommand()

Test method used to see if the X position set by the DrawTo command didn't change because of the Tri command

```
[TestMethod]  
public void XPositionAfterDrawToThenTriangleCommand()
```

## XPositionAfterMoveToThenTriangleCommand()

Test method used to see if the X position set by the MoveTo command didn't change because of the Tri command

```
[TestMethod]  
public void XPositionAfterMoveToThenTriangleCommand()
```

## XPositionAfterTriangleCommand()

Test method to see if the X position didn't change after calling the Triangle method.

```
[TestMethod]  
public void XPositionAfterTriangleCommand()
```

## YPositionAfterDrawToThenTriangleCommand()

Test method used to see if the Y position set by the DrawTo command didn't change because of the Tri command

```
[TestMethod]  
public void YPositionAfterDrawToThenTriangleCommand()
```

## YPositionAfterMoveToThenTriangleCommand()

Test method used to see if the Y position set by the MoveTo command didn't change because of the Tri command



```
[TestMethod]  
public void YPositionAfterMoveToThenTriangleCommand()
```

## YPositionAfterTriangleCommand()

Test method to see if the Y position didn't change after calling the Triangle method.

```
[TestMethod]  
public void YPositionAfterTriangleCommand()
```