Namespace ASE_Assignment_Demo

Classes

AppCanvas

Provides an implementation of the BOOSE.ICanvas interface for performing drawing operations such as shapes, text, and lines on a graphical canvas.

AppCommandFactory

Represents a factory class for creating specific command objects based on the command type. Inherits from the BOOSE.CommandFactory class and overrides the MakeCommand(string) method to return specific command instances based on the provided command type.

AppInt

Represents an extended implementation of the BOOSE.Int class, designed for use in the ASE Assignment application.

AppReset

The `AppReset` class is a command that resets the canvas. It inherits from the `CanvasCommand` class.

AppWrite

Represents a command to write text on canvas.

ProgramInterface

Represents the graphical user interface for the application. Provides methods to interact with the BOOSE framework, parse programs, and handle user input.

Class AppCanvas

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

Provides an implementation of the BOOSE.ICanvas interface for performing drawing operations such as shapes, text, and lines on a graphical canvas.

```
public class AppCanvas : ICanvas
```

Inheritance

object ← AppCanvas

Implements

ICanvas

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

AppCanvas()

Initializes a new instance of the AppCanvas class with default canvas size and settings.

```
public AppCanvas()
```

Properties

Pen

Gets or sets the current drawing pen.

```
public Pen Pen { get; set; }
```

Property Value

Pen

PenColour

Gets or sets the pen color used for drawing.

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

Property Value

object

PenSize

Gets or sets the size of the pen used for drawing.

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

Property Value

int

Xpos

Gets or sets the X-coordinate of the current pen position.

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

Property Value

int

Ypos

Gets or sets the Y-coordinate of the current pen position.

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

Property Value

int

Methods

Circle(int, bool)

Draws a circle at the current pen position with the specified radius.

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

Parameters

`radius` int

The radius of the circle.

`filled` bool

Specifies whether the circle should be filled.

Exceptions

CanvasException

Thrown if the radius is negative or the graphics context is uninitialized.

Clear()

Clears the canvas and resets the pen position to the origin.

```
public void Clear()
```

Dispose()

Disposes of the resources used by the AppCanvas, including the bitmap and graphics context.

```
public void Dispose()
```

DrawTo(int, int)

Draws a line from current pen position to the specified coordinates.

```
public void DrawTo(int toX, int toY)
```

Parameters

`toX` int

The X coordinate to draw to.

`toY` int

The Y coordinate to draw to.

Exceptions

CanvasException

Thrown when the destination coordinates are outside the canvas bounds or graphics context is not initialized.

InitializeGraphics(Bitmap)

Initializes the graphics context with a new bitmap.

```
public void InitializeGraphics(Bitmap bitmap)
```

Parameters

`bitmap` Bitmap

The bitmap to use for the graphics context.

Exceptions

CanvasException

Thrown when graphics initialization fails.

MoveTo(int, int)

Moves the pen to the specified position without drawing.

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

Parameters

x int

The X-coordinate to move to.

y int

The Y-coordinate to move to.

Exceptions

CanvasException

Thrown when the destination coordinates are outside the canvas bounds.

Rect(int, int, bool)

Draws a rectangle at the current pen position with the specified width, height, and filling option.

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

Parameters

width int

The width of the rectangle.

`height` int

The height of the rectangle.

`filled` bool

True to fill the rectangle, false to draw only the outline.

Exceptions

CanvasException

Thrown when the width or height is less than 0 or graphics context is not initialized.

Reset()

Resets the pen position to the top-left corner of the canvas.

```
public void Reset()
```

Set(int, int)

Sets the size of the canvas and reinitializes the graphics context.

```
public void Set(int xsize, int ysize)
```

Parameters

`xsize` int

The width of the canvas.

`ysize` int

The height of the canvas.

SetColour(int, int, int)

Sets the pen color using RGB values.

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

Parameters

`red` int

The red component of the color (0-255).

`green` int

The green component of the color (0-255).

`blue` int

The blue component of the color (0-255).

Exceptions

CanvasException

Thrown when the RGB values are outside the valid range (0-255).

Tri(int, int)

Draws a triangle with the specified base width and height.

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

Parameters

`width` int

The base width of the triangle.

`height` int

The height of the triangle.

Exceptions

CanvasException

Thrown if dimensions are invalid.

WriteText(string)

Writes the specified text at the current pen position.

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

Parameters

`text` string

The text to write on the canvas.

Exceptions

CanvasException

Thrown when the text is null or empty, or the graphics context is not initialized.

getBitmap()

Gets the bitmap representing the canvas.

```
public object getBitmap()
```

Returns

object

The current bitmap of the canvas.

Class AppCommandFactory

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

Represents a factory class for creating specific command objects based on the command type. Inherits from the BOOSE.CommandFactory class and overrides the MakeCommand(string) method to return specific command instances based on the provided command type.

```
public class AppCommandFactory : CommandFactory, ICommandFactory
```

Inheritance

object ← CommandFactory ← AppCommandFactory

Implements

ICommandFactory

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

AppCommandFactory()

Initializes a new instance of the AppCommandFactory class.

```
public AppCommandFactory()
```

Methods

MakeCommand(string)

Creates and returns an BOOSE.ICommand object based on the provided command type. The command type is case-insensitive and is used to determine which specific command to instantiate.

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

Parameters

`commandType` string

The type of the command as a string (e.g., "tri", "write", "clear").

Returns

ICommand

An BOOSE.ICommand object corresponding to the given command type, or a base command if no match is found.

Class AppInt

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

Represents an extended implementation of the BOOSE.Int class, designed for use in the ASE Assignment application.

```
public class AppInt : Int, ICommand
```

Inheritance

object ← Command ← Evaluation ← Int ← AppInt

Implements

ICommand

Inherited Members

Int.Compile() , Int.Execute() , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.CheckParameters(string[]) , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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

Restrictions()

Overrides the BOOSE.Int.Restrictions() method. Removes restrictions on the integer variable functionality.

```
public override void Restrictions()
```

Class AppReset

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

The `AppReset` class is a command that resets the canvas. It inherits from the `CanvasCommand` class.

```
public class AppReset : CanvasCommand, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← AppReset

Implements

ICommand

Inherited Members

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)

Constructors

AppReset()

Initializes a new instance of the AppReset class. This constructor calls the base class constructor without parameters.

```
public AppReset()
```

AppReset(ICanvas)

Initializes a new instance of the AppReset class. This constructor allows you to provide a canvas object, which will be passed to the base class constructor.

```
public AppReset(ICanvas c)
```

Parameters

`c` ICanvas

The ICanvas object that represents the canvas to be reset.

Methods

CheckParameters(string[])

Checks the parameters passed to the `AppReset` command. An exception is thrown indicating that the command does not accept any parameters.

```
public override void CheckParameters(string[] parameter)
```

Parameters

`parameter` string[]

An array of strings representing the parameters passed to the command.

Exceptions

ArgumentException

Thrown when more than one parameter is provided.

Execute()

Executes the reset command on the canvas. If the canvas is not null, it will call the `Reset` method on the canvas object. If the canvas is null, it will output a message indicating that the canvas is not set.

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

Class AppWrite

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

Represents a command to write text on canvas.

```
public class AppWrite : CommandOneParameter, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← CommandOneParameter ← AppWrite

Implements

ICommand

Inherited Members

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)

Constructors

AppWrite()

Initializes a new instance of the AppWrite class.

```
public AppWrite()
```

AppWrite(AppCanvas)

Initializes a new instance of the AppWriteclass with the specified canvas.

```
public AppWrite(AppCanvas canvas)
```

Parameters

`canvas` AppCanvas

The canvas on which the text will be written.

Methods

CheckParameters(string[])

Validates the parameters for the write command.

```
public override void CheckParameters(string[] parameterList)
```

Parameters

`parameterList` string[]

An array of string parameters, where the first parameter is the text to write.

Exceptions

CommandException

Thrown if the text parameter is null, empty, or contains only whitespace.

Execute()

Executes the command to write text on the canvas.

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

Exceptions

CommandException

Thrown if the text to write is null or empty, or if the canvas is not an instance of AppCanvas.

Class ProgramInterface

Namespace: ASE_Assignment_Demo

Assembly: ASE Assignment Demo.dll

Represents the graphical user interface for the application. Provides methods to interact with the BOOSE framework, parse programs, and handle user input.

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

Inheritance

object ← MarshalByRefObject ← Component ← Control ← ScrollableControl ← ContainerControl ← Form ← ProgramInterface

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.OnGotFocus(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.ScaleMinMaxSize(float, float, 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.OnMove(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.DoDragDrop(object, DragDropEffects, Bitmap, Point, bool) ,
Control.DrawToBitmap(Bitmap, Rectangle) , Control.EndInvoke(IAsyncResult) , Control.FindForm() ,
Control.GetTopLevel() , Control.RaiseKeyEvent(object, KeyEventArgs) ,
Control.RaiseMouseEvent(object, MouseEventArgs) , Control.Focus() ,
Control.FromChildHandle(nint) , Control.FromHandle(nint) ,
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.OnDataContextChanged(EventArgs) ,
Control.OnDockChanged(EventArgs) , Control.OnForeColorChanged(EventArgs) ,
Control.OnNotifyMessage(Message) , Control.OnParentBackColorChanged(EventArgs) ,
Control.OnParentBackgroundImageChanged(EventArgs) ,
Control.OnParentBindingContextChanged(EventArgs) , Control.OnParentCursorChanged(EventArgs) ,
Control.OnParentDataContextChanged(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.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.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(nint, 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.DataContext , 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.DataContextChanged , 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

ProgramInterface()

Initializes a new instance of the ProgramInterface class. Sets up the canvas, command factory, program storage, and parser.

```
public ProgramInterface()
```

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.

Namespace ASE_Assignment_Demo.Commands

Classes

AppCall

Represents a command that is part of a compound command, specifically designed for handling method calls. The AppCall class is used to invoke a specific method in a program.

AppCircle

Represents a command that draws a circle on a canvas. The circle's radius is provided as a parameter. This class inherits from BOOSE.CommandOneParameter and is responsible for validating parameters and executing the command to draw the circle on the given canvas.

AppClear

The `AppClear` class is a command that clears the canvas. It inherits from the `CanvasCommand` class.

AppDrawto

Represents a command that draws a line to a specified coordinate (x, y) on the canvas. The command uses two parameters: x and y coordinates for the drawing operation. Inherits from BOOSE.Command TwoParameters to handle two input parameters.

AppElse

Represents the "else" command in a conditional structure, typically used in an "if-else" construct. The AppElse class handles the logic for the "else" branch in such constructs.

AppFor

Represents a command for a "for" loop structure. The AppFor class is used to manage the behavior of a for loop in the program.

AppIf

Represents a command that defines an "If" block in a program. This class is a part of compound commands that are used for conditional execution based on a given condition.

AppRectangle

Represents a command to draw a rectangle on a canvas. This command requires two parameters: width and height, and draws a non-filled rectangle if the parameters are valid.

AppTriangle

Represents a command to draw a triangle on the canvas.

AppWhile

Represents a command that is part of a compound command, specifically designed to handle "while" loop behavior. This class allows for managing the logic related to looping or repeating commands under specific conditions.

Class AppCall

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command that is part of a compound command, specifically designed for handling method calls. The AppCall class is used to invoke a specific method in a program.

```
public class AppCall : Call, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← CompoundCommand ← Call ← AppCall

Implements

ICommand

Inherited Members

Call.methodName , Call.Compile() , Call.Execute() , CompoundCommand.ReduceRestrictions() , CompoundCommand.CheckParameters(string[]) , CompoundCommand.CorrespondingCommand , ConditionalCommand.endLineNumber , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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)

Constructors

AppCall()

Initializes a new instance of the AppCall class. The constructor invokes ReduceRestrictions to remove any restrictions.

```
public AppCall()
```

Methods

Restrictions()

Overrides the Restrictions method to remove any restrictions on the AppCall command. This method can be further implemented to manage specific logic related to restrictions if needed.

```
public override void Restrictions()
```

Class AppCircle

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command that draws a circle on a canvas. The circle's radius is provided as a parameter. This class inherits from BOOSE.CommandOneParameter and is responsible for validating parameters and executing the command to draw the circle on the given canvas.

```
public class AppCircle : CommandOneParameter, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← CommandOneParameter ← AppCircle

Implements

ICommand

Inherited Members

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)

Constructors

AppCircle()

Initializes a new instance of the AppCircle class. This constructor does not initialize the radius.

```
public AppCircle()
```

AppCircle(Canvas, int)

Initializes a new instance of the AppCircle class with the specified canvas and radius.

```
public AppCircle(Canvas canvas, int radius)
```

Parameters

`canvas` Canvas

The canvas where the circle will be drawn.

`radius` int

The radius of the circle.

Methods

CheckParameters(string[])

Checks the validity of the provided parameters for the circle command. The method ensures that exactly one parameter is provided and that it is a valid positive integer for the radius.

```
public override void CheckParameters(string[] parameterList)
```

Parameters

`parameterList` string[]

The list of parameters to check.

Exceptions

ArgumentOutOfRangeException

Thrown when the radius is not a positive integer.

CommandException

Thrown when the number of parameters is incorrect or the radius is invalid.

Execute()

Executes the circle drawing command. This method checks if the radius exceeds 2000, and if so, throws a BOOSE.RestrictionException. It then draws the circle on the canvas.

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

Exceptions

RestrictionException

Thrown when the radius exceeds 2000.

Class AppClear

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

The `AppClear` class is a command that clears the canvas. It inherits from the `CanvasCommand` class.

```
public class AppClear : CanvasCommand, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← AppClear

Implements

ICommand

Inherited Members

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)

Constructors

AppClear()

Initializes a new instance of the AppClear class. This constructor calls the base class constructor without parameters.

```
public AppClear()
```

AppClear(ICanvas)

Initializes a new instance of the AppClear class. This constructor allows you to provide a canvas object, which will be passed to the base class constructor.

```
public AppClear(ICanvas c)
```

Parameters

`c` ICanvas

The ICanvas object that represents the canvas to be cleared.

Methods

CheckParameters(string[])

Checks the parameters passed to the `AppClear` command. An exception is thrown indicating that the command does not accept any parameters.

```
public override void CheckParameters(string[] parameter)
```

Parameters

`parameter` string[]

An array of strings representing the parameters passed to the command.

Exceptions

ArgumentException

Thrown when more than one parameter is provided.

Execute()

Executes the `Clear` command on the canvas. If the canvas is not null, it will call the `Clear` method on the canvas object. If the canvas is null, it will output a message indicating that the canvas is not set.

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

Class AppDrawto

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command that draws a line to a specified coordinate (x, y) on the canvas. The command uses two parameters: x and y coordinates for the drawing operation. Inherits from BOOSE.CommandTwoParameters to handle two input parameters.

```
public class AppDrawto : CommandTwoParameters, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← CommandOneParameter ← CommandTwoParameters ← AppDrawto

Implements

ICommand

Inherited Members

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)

Constructors

AppDrawto()

Initializes a new instance of the AppDrawto class with default values.

```
public AppDrawto()
```

AppDrawto(Canvas, int, int)

Initializes a new instance of the AppDrawto class with the specified canvas and coordinates.

```
public AppDrawto(Canvas canvas, int x, int y)
```

Parameters

`canvas` Canvas

The canvas on which the drawing operation will take place.

`x` int

The x-coordinate of the point to draw to.

`y` int

The y-coordinate of the point to draw to.

Methods

CheckParameters(string[])

Checks the validity of the input parameters for the drawing operation. Ensures that exactly two parameters are provided and that they are valid non-negative integers.

```
public override void CheckParameters(string[] parameterList)
```

Parameters

`parameterList` string[]

An array of string parameters representing the x and y coordinates.

Exceptions

CommandException

Thrown if the number of parameters is incorrect or if the parameters are not valid integers.

Execute()

Executes the drawing operation to the specified coordinates on the canvas. Throws a BOOSE.Restriction Exception if the coordinates are negative.

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

Exceptions

RestrictionException

Thrown if the coordinates are negative.

Class AppElse

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents the "else" command in a conditional structure, typically used in an "if-else" construct. The AppElse class handles the logic for the "else" branch in such constructs.

```
public class AppElse : Else, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← CompoundCommand ← Else ← AppElse

Implements

ICommand

Inherited Members

Else.CheckParameters(string[]) , Else.Compile() , Else.Execute() , Else.CorrespondingEnd , CompoundCommand.ReduceRestrictions() , CompoundCommand.CorrespondingCommand , ConditionalCommand.endLineNumber , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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

Restrictions()

Overrides the Restrictions method to remove any restrictions on the AppElse command. This method can be extended to add any conditional restrictions if needed in the future.

```
public override void Restrictions()
```

Class AppFor

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command for a "for" loop structure. The AppFor class is used to manage the behavior of a for loop in the program.

```
public class AppFor : For, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← For ← AppFor

Implements

ICommand

Inherited Members

For.Compile() , For.Execute() , For.LoopControlV , For.From , For.To , For.Step , ConditionalCommand.endLineNumber , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.CheckParameters(string[]) , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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

Restrictions()

Overrides the Restrictions method to remove any restrictions on the AppFor command. This method can be further extended to manage specific logic related to the "for" loop restrictions, if needed.

```
public override void Restrictions()
```

Class AppIf

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command that defines an "If" block in a program. This class is a part of compound commands that are used for conditional execution based on a given condition.

```
public class AppIf : If, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← CompoundCommand ← If ← AppIf

Implements

ICommand

Inherited Members

CompoundCommand.ReduceRestrictions() , CompoundCommand.CheckParameters(string[]) , CompoundCommand.Compile() , CompoundCommand.CorrespondingCommand , ConditionalCommand.endLineNumber , ConditionalCommand.Execute() , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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)

Constructors

AppIf()

Initializes a new instance of the AppIf class.

```
public AppIf()
```

Methods

Restrictions()

Overrides the ReduceRestrictions method and removes the restriction

```
public override void Restrictions()
```

Class AppRectangle

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command to draw a rectangle on a canvas. This command requires two parameters: width and height, and draws a non-filled rectangle if the parameters are valid.

```
public class AppRectangle : CommandTwoParameters, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← CommandOneParameter ← CommandTwoParameters ← AppRectangle

Implements

ICommand

Inherited Members

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)

Constructors

AppRectangle()

Initializes a new instance of the AppRectangle class.

```
public AppRectangle()
```

AppRectangle(Canvas, int, int)

Initializes a new instance of the AppRectangle class with a specified canvas, width, and height.

```
public AppRectangle(Canvas canvas, int width, int height)
```

Parameters

`canvas` Canvas

The canvas on which the rectangle will be drawn.

`width` int

The width of the rectangle.

`height` int

The height of the rectangle.

Methods

CheckParameters(string[])

Validates the parameters for drawing the rectangle. Ensures that exactly two parameters are provided, both are positive integers, and within the allowed range.

```
public override void CheckParameters(string[] parameterList)
```

Parameters

`parameterList` string[]

The list of parameters for the rectangle.

Exceptions

CommandException

Thrown if the parameters are invalid.

Execute()

Executes the rectangle drawing command by validating parameters and drawing on the canvas.

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

Exceptions

RestrictionException

Thrown if the width or height exceeds 2000.

Class AppTriangle

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command to draw a triangle on the canvas.

```
public class AppTriangle : CommandTwoParameters, ICommand
```

Inheritance

object ← Command ← CanvasCommand ← CommandOneParameter ← CommandTwoParameters ← AppTriangle

Implements

ICommand

Inherited Members

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)

Constructors

AppTriangle()

Initializes a new instance of the AppTriangle class.

```
public AppTriangle()
```

AppTriangle(Canvas, int, int)

Initializes a new instance of the class with the specified canvas, width and height.

```
public AppTriangle(Canvas canvas, int width, int height)
```

Parameters

`canvas` Canvas

The canvas in which the triangle will be drawn.

`width` int

The width of the triangle.

`height` int

The height of the triangle.

Methods

CheckParameters(string[])

Validates the parameters for the triangle command.

```
public override void CheckParameters(string[] parameters)
```

Parameters

`parameters` string[]

An array of strings representing the parameters (width and height) of the triangle.

Remarks

This method checks if the provided parameters are valid (non-null, exactly two values, and both are positive integers).

Exceptions

CommandException

Thrown if the parameters are invalid or not positive integers.

Execute()

Executes the command to draw the triangle on the canvas.

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

Remarks

This method extracts the width and height from the command parameters, validates them, and draws the triangle using the Tri(int, int) method.

Class AppWhile

Namespace: ASE_Assignment_Demo.Commands

Assembly: ASE Assignment Demo.dll

Represents a command that is part of a compound command, specifically designed to handle "while" loop behavior. This class allows for managing the logic related to looping or repeating commands under specific conditions.

```
public class AppWhile : While, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← CompoundCommand ← While ← AppWhile

Implements

ICommand

Inherited Members

CompoundCommand.ReduceRestrictions() , CompoundCommand.CheckParameters(string[]) , CompoundCommand.Compile() , CompoundCommand.CorrespondingCommand , ConditionalCommand.endLineNumber , ConditionalCommand.Execute() , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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)

Constructors

AppWhile()

Initializes a new instance of the AppWhile class. The constructor invokes ReduceRestrictions to remove any restrictions.

```
public AppWhile()
```

Methods

Restrictions()

Overrides the Restrictions method to remove any restrictions on the AppWhile command. This method can be further implemented to manage specific logic related to restrictions if needed.

```
public override void Restrictions()
```

Namespace ASE_Assignment_Demo.Components

Classes

AppArray

Represents a custom implementation of the Array class. This class is used to define an array within the application with modified restriction handling.

AppBoolean

Represents an extended implementation of the Boolean class, designed for use in the ASE Assignment application.

AppEnd

Represents the end of a block of code in the program. This class is part of the compound commands and is used to signify the conclusion of a control structure or code block.

AppReal

Represents an extended implementation of the Real class for handling real number functionality in the ASE Assignment application. This class overrides the Value property and implements a custom restriction system to track the number of restrictions applied.

AppStoredProgram

Represents a stored program that is capable of running a sequence of commands on a given canvas. This class extends the functionality of the StoredProgram class, specifically providing an implementation of the Run method, which executes all the commands sequentially as long as there are commands left.

Class AppArray

Namespace: ASE_Assignment_Demo.Components

Assembly: ASE Assignment Demo.dll

Represents a custom implementation of the Array class. This class is used to define an array within the application with modified restriction handling.

```
public class AppArray : Array, ICommand
```

Inheritance

object ← Command ← Evaluation ← Array ← AppArray

Implements

ICommand

Inherited Members

Array.PEEK , Array.POKE , Array.type , Array.rows , Array.columns , Array.valueInt , Array.valueReal , Array.intArray , Array.realArray , Array.pokeValue , Array.peekVar , Array.rowS , Array.columnS , Array.row , Array.column , Array.ArrayRestrictions() , Array.ReduceRestrictionCounter() , Array.Compile() , Array.CheckParameters(string[]) , Array.Execute() , Array.ProcessArrayParametersCompile(bool) , Array.ProcessArrayParametersExecute(bool) , Array.SetIntArray(int, int, int) , Array.SetRealArray(double, int, int) , Array.GetIntArray(int, int) , Array.GetRealArray(int, int) , Array.Rows , Array.Columns , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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)

Constructors

AppArray()

Initializes a new instance of the AppArray class. The constructor invokes ReduceRestrictionCounter to manage the restriction counter.

```
public AppArray()
```

Class AppBoolean

Namespace: ASE_Assignment_Demo.Components

Assembly: ASE Assignment Demo.dll

Represents an extended implementation of the Boolean class, designed for use in the ASE Assignment application.

```
public class AppBoolean : Boolean, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← AppBoolean

Implements

ICommand

Inherited Members

Boolean.Compile() , Boolean.Execute() , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.CheckParameters(string[]) , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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

Restrictions()

Overrides the Restrictions method. Removes restrictions on the boolean variable functionality.

```
public override void Restrictions()
```

Class AppEnd

Namespace: ASE_Assignment_Demo.Components

Assembly: ASE Assignment Demo.dll

Represents the end of a block of code in the program. This class is part of the compound commands and is used to signify the conclusion of a control structure or code block.

```
public class AppEnd : End, ICommand
```

Inheritance

object ← Command ← Evaluation ← Boolean ← ConditionalCommand ← CompoundCommand ← End ← AppEnd

Implements

ICommand

Inherited Members

End.Compile() , End.Execute() , CompoundCommand.ReduceRestrictions() , CompoundCommand.CheckParameters(string[]) , CompoundCommand.CorrespondingCommand , ConditionalCommand.endLineNumber , ConditionalCommand.EndLineNumber , ConditionalCommand.Condition , ConditionalCommand.LineNumber , ConditionalCommand.CondType , ConditionalCommand.ReturnLineNumber , Boolean.BoolValue , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Value , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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

Restrictions()

Overrides the Restrictions method to remove any restrictions on the AppEnd command.

```
public override void Restrictions()
```

Class AppReal

Namespace: ASE_Assignment_Demo.Components

Assembly: ASE Assignment Demo.dll

Represents an extended implementation of the Real class for handling real number functionality in the ASE Assignment application. This class overrides the Value property and implements a custom restriction system to track the number of restrictions applied.

```
public class AppReal : Real, ICommand
```

Inheritance

object ← Command ← Evaluation ← Real ← AppReal

Implements

ICommand

Inherited Members

Real.Compile() , Real.Execute() , Evaluation.expression , Evaluation.evaluatedExpression , Evaluation.varName , Evaluation.value , Evaluation.CheckParameters(string[]) , Evaluation.ProcessExpression(string) , Evaluation.Expression , Evaluation.VarName , Evaluation.Local , Command.program , Command.parameterList , Command.parameters , Command.paramsint , Command.Set(StoredProgram, string) , 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)

Constructors

AppReal()

Initializes a new instance of the AppReal class and calls the Restrictions method to apply any restrictions.

```
public AppReal()
```

Properties

Value

Gets or sets the value of the real number. Overrides the base Value property to access the private field.

```
public double Value { get; set; }
```

Property Value

double

Methods

Restrictions()

Implements the Restrictions method to enforce a restriction counter. Throws an exception if the restriction limit of 50 is exceeded.

```
public override void Restrictions()
```

Exceptions

RestrictionException

Thrown if the restriction count exceeds 50.

Class AppStoredProgram

Namespace: ASE_Assignment_Demo.Components

Assembly: ASE Assignment Demo.dll

Represents a stored program that is capable of running a sequence of commands on a given canvas. This class extends the functionality of the StoredProgram class, specifically providing an implementation of the Run method, which executes all the commands sequentially as long as there are commands left.

```
public class AppStoredProgram : StoredProgram, IList, ICollection, IEnumerable,
ICloneable, IStoredProgram
```

Inheritance

object ← ArrayList ← StoredProgram ← AppStoredProgram

Implements

IList, ICollection, IEnumerable, ICloneable, IStoredProgram

Inherited Members

StoredProgram.SyntaxOk , StoredProgram.AddMethod(Method) , StoredProgram.GetMethod(string) , StoredProgram.AddVariable(Evaluation) , StoredProgram.GetVariable(string) , StoredProgram.GetVariable(int) , StoredProgram.FindVariable(Evaluation) , StoredProgram.FindVariable(string) , StoredProgram.VariableExists(string) , StoredProgram.GetVarValue(string) , StoredProgram.UpdateVariable(string, int) , StoredProgram.UpdateVariable(string, double) , StoredProgram.UpdateVariable(string, bool) , StoredProgram.DeleteVariable(string) , StoredProgram.IsExpression(string) , StoredProgram.EvaluateExpressionWithString(string) , StoredProgram.EvaluateExpression(string) , StoredProgram.Push(ConditionalCommand) , StoredProgram.Pop() , StoredProgram.Add(Command) , StoredProgram.NextCommand() , StoredProgram.ResetProgram() , StoredProgram.Commandsleft() , StoredProgram.PC , ArrayList.Adapter(IList) , ArrayList.Add(object) , ArrayList.AddRange(ICollection) , ArrayList.BinarySearch(int, int, object, IComparer) , ArrayList.BinarySearch(object) , ArrayList.BinarySearch(object, IComparer) , ArrayList.Clear() , ArrayList.Clone() , ArrayList.Contains(object) , ArrayList.CopyTo(Array) , ArrayList.CopyTo(Array, int) , ArrayList.CopyTo(int, Array, int, int) , ArrayList.FixedSize(ArrayList) , ArrayList.FixedSize(IList) , ArrayList.GetEnumerator() , ArrayList.GetEnumerator(int, int) , ArrayList.GetRange(int, int) , ArrayList.IndexOf(object) , ArrayList.IndexOf(object, int) , ArrayList.IndexOf(object, int, int) , ArrayList.Insert(int, object) , ArrayList.InsertRange(int, ICollection) , ArrayList.LastIndexOf(object) , ArrayList.LastIndexOf(object, int) , ArrayList.LastIndexOf(object, int, int) , ArrayList.ReadOnly(ArrayList) , ArrayList.ReadOnly(IList) , ArrayList.Remove(object) ,

ArrayList.RemoveAt(int) , ArrayList.RemoveRange(int, int) , ArrayList.Repeat(object, int) , ArrayList.Reverse() , ArrayList.Reverse(int, int) , ArrayList.SetRange(int, ICollection) , ArrayList.Sort() , ArrayList.Sort(IComparer) , ArrayList.Sort(int, int, IComparer) , ArrayList.Synchronized(ArrayList) , ArrayList.Synchronized(IList) , ArrayList.ToArray() , ArrayList.ToArray(Type) , ArrayList.TrimToSize() , ArrayList.Capacity , ArrayList.Count , ArrayList.IsFixedSize , ArrayList.IsReadOnly , ArrayList.IsSynchronized , ArrayList.this[int] , ArrayList.SyncRoot , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

AppStoredProgram(ICanvas)

Initializes a new instance of the AppStoredProgram class with the given canvas.

```
public AppStoredProgram(ICanvas canvas)
```

Parameters

`canvas` ICanvas

The canvas on which the stored program will operate.

Methods

Run()

Runs the stored program by executing each command in sequence. The method loops and executes commands until no more commands are left.

```
public override void Run()
```

Namespace ASE_Assignment_Demo.File Operation

Classes

LoadCanvas

Represents a class responsible for loading a canvas image from a file. The LoadCanvas class allows the user to select an image file (PNG, JPEG, BMP) using a file dialog and loads the image as a bitmap.

ReadCommand

Represents a class responsible for reading commands from a text file. The ReadCommand class allows the user to select a text file (with .txt extension) using a file dialog and loads its content as a string.

SaveCanvas

Represents a class responsible for saving the canvas image to a file. The SaveCanvas class allows the user to save the canvas image in different formats (PNG, JPEG, BMP) using a file dialog to select the save location.

WriteCommand

Represents a class responsible for writing command data to a file. The WriteCommand class allows users to save commands to a text file using a file dialog for selecting the save location.

Class LoadCanvas

Namespace: ASE_Assignment_Demo.FileOperation

Assembly: ASE Assignment Demo.dll

Represents a class responsible for loading a canvas image from a file. The LoadCanvas class allows the user to select an image file (PNG, JPEG, BMP) using a file dialog and loads the image as a bitmap.

```
public class LoadCanvas
```

Inheritance

object ← LoadCanvas

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

LoadCanvas()

Initializes a new instance of the LoadCanvas class.

```
public LoadCanvas()
```

Methods

Load()

Opens a file dialog for the user to select an image file to load as a bitmap. The supported formats are PNG, JPEG, and BMP.

```
public Bitmap Load()
```

Returns

Bitmap

Returns a Bitmap representing the loaded image, or null if the operation was canceled or failed.

Class ReadCommand

Namespace: ASE_Assignment_Demo.FileOperation

Assembly: ASE Assignment Demo.dll

Represents a class responsible for reading commands from a text file. The ReadCommand class allows the user to select a text file (with .txt extension) using a file dialog and loads its content as a string.

```
public class ReadCommand
```

Inheritance

object ← ReadCommand

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

ReadCommand()

Initializes a new instance of the ReadCommand class.

```
public ReadCommand()
```

Methods

Read()

Opens a file dialog for the user to select a text file and reads the content of the file.

```
public string Read()
```

Returns

string

Returns the content of the file as a string. If the operation fails or is canceled, an empty string is returned.

Class SaveCanvas

Namespace: ASE_Assignment_Demo.FileOperation

Assembly: ASE Assignment Demo.dll

Represents a class responsible for saving the canvas image to a file. The SaveCanvas class allows the user to save the canvas image in different formats (PNG, JPEG, BMP) using a file dialog to select the save location.

```
public class SaveCanvas
```

Inheritance

object ← SaveCanvas

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

SaveCanvas()

Initializes a new instance of the SaveCanvas class.

```
public SaveCanvas()
```

Methods

Save(Bitmap)

Saves the provided bitmap image of the canvas to a file using a save file dialog. The user can choose between different image formats like PNG, JPEG, or BMP.

```
public void Save(Bitmap canvasBitmap)
```

Parameters

`canvasBitmap` Bitmap

The bitmap representing the canvas image to be saved.

Class WriteCommand

Namespace: ASE_Assignment_Demo.FileOperation

Assembly: ASE Assignment Demo.dll

Represents a class responsible for writing command data to a file. The WriteCommand class allows users to save commands to a text file using a file dialog for selecting the save location.

```
public class WriteCommand
```

Inheritance

object ← WriteCommand

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

WriteCommand()

Initializes a new instance of the WriteCommand class.

```
public WriteCommand()
```

Methods

Write(string)

Writes the provided command string to a file using a save file dialog. The user can select the location to save the file, and the content is saved as a text file.

```
public void Write(string Commands)
```

Parameters

`Commands` [string](#)

The string containing the commands to be saved.