Namespace Nodify

Classes

AllGestures

All gestures must match.

AnyGesture

At least one gesture must match.

BaseConnection

Represents the base class for shapes that are drawn from a Source point to a Target point.

BoxValue

CircuitConnection

Represents a line that is controlled by an angle.

Connection

Represents a cubic bezier curve.

ConnectionEventArgs

Provides data for BaseConnection related routed events.

Connector

Represents a connector control that can start and complete a PendingConnection. Has a Element Connector that the Anchor is calculated from for the PendingConnection. Center of this control is used if missing.

ConnectorEventArgs

Provides data for Connector related routed events.

ContainerDefaultState

The default state of the ItemContainer.

ContainerDraggingState

Dragging state of the container.

ContainerState

The base class for container states.

DecoratorContainer

The container for all the items generated from the Decorators collection.

EditorCommands

EditorDefaultState

The default state of the editor.

Default State
- mouse left down -> Selecting State
- mouse right down -> Panning State

Selecting State
- mouse left up -> Default State
- mouse right down -> Panning State

Panning State
- mouse right up -> previous state (Selecting State or Default State)
- mouse left up -> Panning State

EditorGestures

Gestures used by built-in controls inside the NodifyEditor.

EditorGestures.ConnectionGestures

Gestures used by the BaseConnection.

EditorGestures.ConnectorGestures

Gestures used by the Connector.

EditorGestures.GroupingNodeGestures

Gestures for the GroupingNode.

EditorGestures.ItemContainerGestures

Gestures for the item containers.

EditorGestures.NodifyEditorGestures

Gestures for the editor.

EditorGestures.SelectionGestures

Gestures for the selection.

EditorPanningState

The panning state of the editor.

EditorSelectingState

The selecting state of the editor.

EditorState

The base class for editor states.

GroupingNode

Defines a panel with a header that groups ItemContainers inside it and can be resized.

InputGestureRef

An input gesture that allows changing its logic at runtime without changing its reference. Useful for classes that capture the object reference without the posibility of updating it. (e.g. EditorCommands)

ItemContainer

The container for all the items generated by the ItemsSource of the NodifyEditor.

KnotNode

Represents a control that owns a Connector.

LineConnection

Represents a line that has an arrow indicating its Direction.

MultiGesture

Combines multiple input gestures.

Node

Represents a control that has a list of InputConnectors and a list of OutputConnectors.

NodeInput

Represents the default control for the InputConnectorTemplate.

NodeOutput

Represents the default control for the OutputConnectorTemplate.

NodifyCanvas

A canvas like panel that works with INodifyCanvasItems.

NodifyEditor

Groups ItemContainers and Connections in an area that you can drag, zoom and select.

PendingConnection

Represents a pending connection usually started by a Connector which invokes the Completed Command when completed.

PendingConnectionEventArgs

Provides data for PendingConnection related routed events.

ResizeEventArgs

Provides data for resize related routed events.

SelectionHelper

Helps with selecting ItemContainers and updating the SelectedArea and IsSelecting properties.

StateNode
Represents a control that acts as a Connector.

Interfaces

INodifyCanvasItem
Interface for items inside a NodifyCanvas.

Enums

ArrowHeadEnds
The end at which the arrow head is drawn.

ArrowHeadShape
The shape of the arrowhead.

ConnectionDirection
The direction in which a connection is oriented.

ConnectionOffsetMode
Specifies the offset type that can be applied to a BaseConnection using the SourceOffset and the TargetOffset values.

EditorCommands.Alignment
Specifies the possible alignment values used by the Align command.

GroupingMovementMode
Specifies the possible movement modes of a GroupingNode.

MultiGesture.Match
The strategy used by Matches(object, InputEventArgs).

SelectionHelper.SelectionType
Available selection logic.

Delegates

ConnectionEventHandler
Represents the method that will handle BaseConnection related routed events.

ConnectorEventHandler
Represents the method that will handle Connector related routed events.

PendingConnectionEventHandler

Represents the method that will handle PendingConnection related routed events.

PreviewLocationChanged

Delegate used to notify when an ItemContainer is previewing a new location.

ResizeEventHandler

Represents the method that will handle resize related routed events.

Class AllGestures

Namespace: Nodify

Assembly: Nodify.dll

All gestures must match.

```
public sealed class AllGestures : MultiGesture
```

Inheritance

object ← InputGesture ← MultiGesture ← AllGestures

Inherited Members

MultiGesture.None , MultiGesture.Matches(object, InputEventArgs)

Constructors

AllGestures(params InputGesture[])

```
public AllGestures(params InputGesture[] gestures)
```

Parameters

`gestures` InputGesture[]

Class AnyGesture

Namespace: Nodify

Assembly: Nodify.dll

At least one gesture must match.

```
public sealed class AnyGesture : MultiGesture
```

Inheritance

object ← InputGesture ← MultiGesture ← AnyGesture

Inherited Members

MultiGesture.None , MultiGesture.Matches(object, InputEventArgs)

Constructors

AnyGesture(params InputGesture[])

```
public AnyGesture(params InputGesture[] gestures)
```

Parameters

`gestures` InputGesture[]

Enum ArrowHeadEnds

Namespace: Nodify

Assembly: Nodify.dll

The end at which the arrow head is drawn.

```
public enum ArrowHeadEnds
```

Fields

`Both = 2`

Arrow heads at both ends.

`End = 1`

Arrow head at end.

`None = 3`

No arrow head.

`Start = 0`

Arrow head at start.

Enum ArrowHeadShape

Namespace: Nodify

Assembly: Nodify.dll

The shape of the arrowhead.

```
public enum ArrowHeadShape
```

Fields

`Arrowhead = 0`

The default arrowhead.

`Ellipse = 1`

An ellipse.

`Rectangle = 2`

A rectangle.

Class BaseConnection

Namespace: Nodify

Assembly: Nodify.dll

Represents the base class for shapes that are drawn from a Source point to a Target point.

```
public abstract class BaseConnection : Shape
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Shape ← BaseConnection

Derived

Connection, LineConnection

Fields

ArrowEndsProperty

```
public static readonly DependencyProperty ArrowEndsProperty
```

Field Value

DependencyProperty

ArrowShapeProperty

```
public static readonly DependencyProperty ArrowShapeProperty
```

Field Value

DependencyProperty

ArrowSizeProperty

```
public static readonly DependencyProperty ArrowSizeProperty
```

Field Value

DependencyProperty

DirectionProperty

```
public static readonly DependencyProperty DirectionProperty
```

Field Value

DependencyProperty

DirectionalArrowsCountProperty

```
public static readonly DependencyProperty DirectionalArrowsCountProperty
```

Field Value

DependencyProperty

DirectionalArrowsOffsetProperty

```
public static readonly DependencyProperty DirectionalArrowsOffsetProperty
```

Field Value

DependencyProperty

DisconnectCommandProperty

```
public static readonly DependencyProperty DisconnectCommandProperty
```

Field Value

DependencyProperty

DisconnectEvent

```
public static readonly RoutedEvent DisconnectEvent
```

Field Value

RoutedEvent

FontFamilyProperty

```
public static readonly DependencyProperty FontFamilyProperty
```

Field Value

DependencyProperty

FontSizeProperty

```
public static readonly DependencyProperty FontSizeProperty
```

Field Value

DependencyProperty

FontStretchProperty

```
public static readonly DependencyProperty FontStretchProperty
```

Field Value

DependencyProperty

FontStyleProperty

```
public static readonly DependencyProperty FontStyleProperty
```

Field Value

DependencyProperty

FontWeightProperty

```
public static readonly DependencyProperty FontWeightProperty
```

Field Value

DependencyProperty

ForegroundProperty

```
public static readonly DependencyProperty ForegroundProperty
```

Field Value

DependencyProperty

SourceOffsetModeProperty

```
public static readonly DependencyProperty SourceOffsetModeProperty
```

Field Value

DependencyProperty

SourceOffsetProperty

```
public static readonly DependencyProperty SourceOffsetProperty
```

Field Value

DependencyProperty

SourceOrientationProperty

```
public static readonly DependencyProperty SourceOrientationProperty
```

Field Value

DependencyProperty

SourceProperty

```
public static readonly DependencyProperty SourceProperty
```

Field Value

DependencyProperty

SpacingProperty

```
public static readonly DependencyProperty SpacingProperty
```

Field Value

DependencyProperty

SplitCommandProperty

```
public static readonly DependencyProperty SplitCommandProperty
```

Field Value

DependencyProperty

SplitEvent

```
public static readonly RoutedEvent SplitEvent
```

Field Value

RoutedEvent

TargetOffsetModeProperty

```
public static readonly DependencyProperty TargetOffsetModeProperty
```

Field Value

DependencyProperty

TargetOffsetProperty

```
public static readonly DependencyProperty TargetOffsetProperty
```

Field Value

DependencyProperty

TargetOrientationProperty

```
public static readonly DependencyProperty TargetOrientationProperty
```

Field Value

DependencyProperty

TargetProperty

```
public static readonly DependencyProperty TargetProperty
```

Field Value

DependencyProperty

TextProperty

```
public static readonly DependencyProperty TextProperty
```

Field Value

DependencyProperty

ZeroVector

Gets a vector that has its coordinates set to 0.

```
protected static readonly Vector ZeroVector
```

Field Value

Vector

Properties

ArrowEnds

Gets or sets the arrowhead ends.

```
public ArrowHeadEnds ArrowEnds { get; set; }
```

Property Value

ArrowHeadEnds

ArrowShape

Gets or sets the arrowhead ends.

```
public ArrowHeadShape ArrowShape { get; set; }
```

Property Value

ArrowHeadShape

ArrowSize

Gets or sets the size of the arrow head.

```
public Size ArrowSize { get; set; }
```

Property Value

Size

DefiningGeometry

Gets a value that represents the Geometry of the Shape.

```
protected override Geometry DefiningGeometry { get; }
```

Property Value

Geometry

The Geometry of the Shape.

Direction

Gets or sets the direction in which this connection is flowing.

```
public ConnectionDirection Direction { get; set; }
```

Property Value

ConnectionDirection

DirectionalArrowsCount

Gets or sets the number of arrows to be drawn on the line in the direction of the connection (see Direction).

```
public uint DirectionalArrowsCount { get; set; }
```

Property Value

uint

DirectionalArrowsOffset

Gets or sets the offset of the arrows drawn by the DirectionalArrowsCount (value is clamped between 0 and 1).

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

Property Value

double

DisconnectCommand

Removes this connection. Triggered by EditorGestures.Connection.Disconnect gesture. Parameter is the location where the disconnect ocurred.

```
public ICommand? DisconnectCommand { get; set; }
```

Property Value

ICommand

FontFamily

Gets or sets the preferred top-level font family for the content of the element.

```
public FontFamily FontFamily { get; set; }
```

Property Value

FontFamily

> The preferred font family or a primary preferred font family with one or more fallback font families. The default is the font determined by the MessageFontFamily value.

FontSize

Gets or sets the font size for the content of the element.

```
[TypeConverter(typeof(FontSizeConverter))]
public double FontSize { get; set; }
```

Property Value

double

The desired font size to use in device independent pixels, greater than 0.001 and less than or equal to 35791. The default depends on current system settings and depends on the MessageFontSize value.

Exceptions

ArgumentException

FontSize is set to a value greater than 35791 or less than or equal to 0.001.

FontStretch

Gets or sets the font-stretching characteristics for the content of the element.

```
public FontStretch FontStretch { get; set; }
```

Property Value

FontStretch

The desired font-stretching characteristics to use. The default is Normal.

FontStyle

Gets or sets the font style for the content of the element.

```
public FontStyle FontStyle { get; set; }
```

Property Value

FontStyle

The desired font style. The default is determined by the MessageFontStyle value.

FontWeight

Gets or sets the top-level font weight for the content of the element.

```
public FontWeight FontWeight { get; set; }
```

Property Value

FontWeight

The desired font weight. The default value is determined by the MessageFontWeight value.

Foreground

The brush used to render the Text.

```
public Brush? Foreground { get; set; }
```

Property Value

Brush

Source

Gets or sets the start point of this connection.

```
public Point Source { get; set; }
```

Property Value

Point

SourceOffset

Gets or sets the offset from the Source point.

```
public Size SourceOffset { get; set; }
```

Property Value

Size

SourceOffsetMode

Gets or sets the ConnectionOffsetMode to apply to the Source when drawing the connection.

```
public ConnectionOffsetMode SourceOffsetMode { get; set; }
```

Property Value

ConnectionOffsetMode

SourceOrientation

Gets or sets the orientation in which this connection is flowing.

```
public Orientation SourceOrientation { get; set; }
```

Property Value

Orientation

Spacing

The distance between the start point and the where the angle breaks.

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

Property Value

double

SplitCommand

Splits the connection. Triggered by EditorGestures.Connection.Split gesture. Parameter is the location where the splitting ocurred.

```
public ICommand? SplitCommand { get; set; }
```

Property Value

ICommand

Target

Gets or sets the end point of this connection.

```
public Point Target { get; set; }
```

Property Value

Point

TargetOffset

Gets or sets the offset from the Target point.

```
public Size TargetOffset { get; set; }
```

Property Value

Size

TargetOffsetMode

Gets or sets the ConnectionOffsetMode to apply to the Target when drawing the connection.

```
public ConnectionOffsetMode TargetOffsetMode { get; set; }
```

Property Value

ConnectionOffsetMode

TargetOrientation

Gets or sets the orientation in which this connection is flowing.

```
public Orientation TargetOrientation { get; set; }
```

Property Value

Orientation

Text

Gets or sets the text contents of the BaseConnection.

```
public string? Text { get; set; }
```

Property Value

string

Methods

DrawArrowGeometry(StreamGeometryContext, Point, Point, ConnectionDirection, ArrowHeadShape, Orientation)

```
protected virtual void DrawArrowGeometry(StreamGeometryContext context, Point source, Point
target, ConnectionDirection arrowDirection = ConnectionDirection.Forward, ArrowHeadShape
shape = ArrowHeadShape.Arrowhead, Orientation orientation = Orientation.Horizontal)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

`arrowDirection` ConnectionDirection

`shape` ArrowHeadShape

`orientation` Orientation

DrawDefaultArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation)

```
protected virtual void DrawDefaultArrowhead(StreamGeometryContext context, Point source,
Point target, ConnectionDirection arrowDirection = ConnectionDirection.Forward, Orientation
orientation = Orientation.Horizontal)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

`arrowDirection` ConnectionDirection

`orientation` Orientation

DrawDirectionalArrowheadGeometry(StreamGeometryContext, Vector, Point)

```
protected virtual void DrawDirectionalArrowheadGeometry(StreamGeometryContext context,
Vector direction, Point location)
```

Parameters

`context` StreamGeometryContext

`direction` Vector

`location` Point

DrawDirectionalArrowsGeometry(StreamGeometryContext, Point, Point)

```
protected virtual void DrawDirectionalArrowsGeometry(StreamGeometryContext context, Point
source, Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

DrawEllipseArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation)

```
protected virtual void DrawEllipseArrowhead(StreamGeometryContext context, Point source,
Point target, ConnectionDirection arrowDirection = ConnectionDirection.Forward, Orientation
orientation = Orientation.Horizontal)
```

Parameters

`context` StreamGeometryContext

`source` Point

target Point

arrowDirection ConnectionDirection

orientation Orientation

DrawLineGeometry(StreamGeometryContext, Point, Point)

```
protected abstract ((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource,
Point ArrowEndTarget)) DrawLineGeometry(StreamGeometryContext context, Point source,
Point target)
```

Parameters

context StreamGeometryContext

source Point

target Point

Returns

((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource, Point ArrowEnd Target))

DrawRectangleArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation)

```
protected virtual void DrawRectangleArrowhead(StreamGeometryContext context, Point source,
Point target, ConnectionDirection arrowDirection = ConnectionDirection.Forward, Orientation
orientation = Orientation.Horizontal)
```

Parameters

context StreamGeometryContext

source Point

target Point

`arrowDirection` ConnectionDirection

`orientation` Orientation

GetOffset()

Gets the resulting offset after applying the SourceOffsetMode.

```
protected virtual (Vector SourceOffset, Vector TargetOffset) GetOffset()
```

Returns

(Vector SourceOffset, Vector TargetOffset)

GetTextPosition(FormattedText, Point, Point)

```
protected virtual Point GetTextPosition(FormattedText text, Point source, Point target)
```

Parameters

`text` FormattedText

`source` Point

`target` Point

Returns

Point

OnMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseDown(MouseButtonEventArgs e)
```

Parameters

`e` MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

OnMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseUp(MouseButtonEventArgs e)
```

Parameters

`e` MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

OnRender(DrawingContext)

Provides a means to change the default appearance of a Shape element.

```
protected override void OnRender(DrawingContext drawingContext)
```

Parameters

`drawingContext` DrawingContext

A DrawingContext object that is drawn during the rendering pass of this Shape.

StartAnimation(double)

Starts animating the directional arrows.

```
public void StartAnimation(double duration = 1.5)
```

Parameters

`duration` double

The duration for moving an arrowhead from Source to Target.

StopAnimation()

Stops the animation started by StartAnimation(double)

```
public void StopAnimation()
```

Events

Disconnect

Triggered by the EditorGestures.Connection.Disconnect gesture.

```
public event ConnectionEventHandler Disconnect
```

Event Type

ConnectionEventHandler

Split

Triggered by the EditorGestures.Connection.Split gesture.

```
public event ConnectionEventHandler Split
```

Event Type

ConnectionEventHandler

Class BoxValue

Namespace: Nodify

Assembly: Nodify.dll

```
public static class BoxValue
```

Inheritance

object ← BoxValue

Fields

ArrowSize

```
public static readonly object ArrowSize
```

Field Value

object

ConnectionOffset

```
public static readonly object ConnectionOffset
```

Field Value

object

Double0

```
public static readonly object Double0
```

Field Value

object

Double1

```
public static readonly object Double1
```

Field Value

object

Double1000

```
public static readonly object Double1000
```

Field Value

object

Double2

```
public static readonly object Double2
```

Field Value

object

Double45

```
public static readonly object Double45
```

Field Value

object

DoubleHalf

```
public static readonly object DoubleHalf
```

Field Value

object

False

```
public static readonly object False
```

Field Value

object

Int0

```
public static readonly object Int0
```

Field Value

object

Int1

```
public static readonly object Int1
```

Field Value

object

Point

```
public static readonly object Point
```

Field Value

object

Rect

```
public static readonly object Rect
```

Field Value

object

Size

```
public static readonly object Size
```

Field Value

object

Thickness2

```
public static readonly object Thickness2
```

Field Value

object

True

```
public static readonly object True
```

Field Value

object

UInt0

```
public static readonly object UInt0
```

Field Value

object

UInt1

```
public static readonly object UInt1
```

Field Value

object

Class CircuitConnection

Namespace: Nodify

Assembly: Nodify.dll

Represents a line that is controlled by an angle.

```
public class CircuitConnection : LineConnection
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Shape ← BaseConnection ← LineConnection ← CircuitConnection

Inherited Members

LineConnection.DrawDefaultArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , LineConnection.InterpolateLineSegment(Point, Point, double) , BaseConnection.SourceProperty , BaseConnection.TargetProperty , BaseConnection.SourceOffsetProperty , BaseConnection.TargetOffsetProperty , BaseConnection.SourceOffsetModeProperty , BaseConnection.TargetOffsetModeProperty , BaseConnection.SourceOrientationProperty , BaseConnection.TargetOrientationProperty , BaseConnection.DirectionProperty , BaseConnection.DirectionalArrowsCountProperty , BaseConnection.DirectionalArrowsOffsetProperty , BaseConnection.SpacingProperty , BaseConnection.ArrowSizeProperty , BaseConnection.ArrowEndsProperty , BaseConnection.ArrowShapeProperty , BaseConnection.SplitCommandProperty , BaseConnection.DisconnectCommandProperty , BaseConnection.ForegroundProperty , BaseConnection.TextProperty , BaseConnection.FontSizeProperty , BaseConnection.FontFamilyProperty , BaseConnection.FontWeightProperty , BaseConnection.FontStyleProperty , BaseConnection.FontStretchProperty , BaseConnection.Source , BaseConnection.Target , BaseConnection.SourceOffset , BaseConnection.TargetOffset , BaseConnection.SourceOffsetMode , BaseConnection.TargetOffsetMode , BaseConnection.SourceOrientation , BaseConnection.TargetOrientation , BaseConnection.Direction , BaseConnection.DirectionalArrowsCount , BaseConnection.DirectionalArrowsOffset , BaseConnection.ArrowEnds , BaseConnection.ArrowShape , BaseConnection.Spacing , BaseConnection.ArrowSize , BaseConnection.SplitCommand , BaseConnection.DisconnectCommand , BaseConnection.Foreground , BaseConnection.Text , BaseConnection.FontSize , BaseConnection.FontFamily , BaseConnection.FontStyle , BaseConnection.FontWeight , BaseConnection.FontStretch , BaseConnection.DisconnectEvent , BaseConnection.SplitEvent , BaseConnection.Disconnect , BaseConnection.Split , BaseConnection.ZeroVector , BaseConnection.DefiningGeometry ,

BaseConnection.DrawDirectionalArrowheadGeometry(StreamGeometryContext, Vector, Point) , BaseConnection.DrawArrowGeometry(StreamGeometryContext, Point, Point, ConnectionDirection, ArrowHeadShape, Orientation) , BaseConnection.DrawRectangleArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.DrawEllipseArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.GetOffset() , BaseConnection.StartAnimation(double) , BaseConnection.StopAnimation() , BaseConnection.OnMouseDown(MouseButtonEventArgs) , BaseConnection.OnMouseUp(MouseButtonEventArgs) , BaseConnection.OnRender(DrawingContext)

Fields

AngleProperty

```
public static readonly DependencyProperty AngleProperty
```

Field Value

DependencyProperty

Degrees

```
protected const double Degrees = 0.017453292519943295
```

Field Value

double

Properties

Angle

The angle of the connection in degrees.

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

Property Value

double

Methods

DrawDirectionalArrowsGeometry(StreamGeometryContext, Point, Point)

```
protected override void DrawDirectionalArrowsGeometry(StreamGeometryContext context, Point
source, Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

DrawLineGeometry(StreamGeometryContext, Point, Point)

```
protected override ((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource,
Point ArrowEndTarget)) DrawLineGeometry(StreamGeometryContext context, Point source,
Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

Returns

((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource, Point ArrowEndTarget))

GetTextPosition(FormattedText, Point, Point)

```
protected override Point GetTextPosition(FormattedText text, Point source, Point target)
```

Parameters

`text` FormattedText

`source` Point

`target` Point

Returns

Point

Class Connection

Namespace: Nodify

Assembly: Nodify.dll

Represents a cubic bezier curve.

```
public class Connection : BaseConnection
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Shape ← BaseConnection ← Connection

Inherited Members

BaseConnection.SourceProperty , BaseConnection.TargetProperty , BaseConnection.SourceOffsetProperty , BaseConnection.TargetOffsetProperty , BaseConnection.SourceOffsetModeProperty , BaseConnection.TargetOffsetModeProperty , BaseConnection.SourceOrientationProperty , BaseConnection.TargetOrientationProperty , BaseConnection.DirectionProperty , BaseConnection.DirectionalArrowsCountProperty , BaseConnection.DirectionalArrowsOffsetProperty , BaseConnection.SpacingProperty , BaseConnection.ArrowSizeProperty , BaseConnection.ArrowEndsProperty , BaseConnection.ArrowShapeProperty , BaseConnection.SplitCommandProperty , BaseConnection.DisconnectCommandProperty , BaseConnection.ForegroundProperty , BaseConnection.TextProperty , BaseConnection.FontSizeProperty , BaseConnection.FontFamilyProperty , BaseConnection.FontWeightProperty , BaseConnection.FontStyleProperty , BaseConnection.FontStretchProperty , BaseConnection.Source , BaseConnection.Target , BaseConnection.SourceOffset , BaseConnection.TargetOffset , BaseConnection.SourceOffsetMode , BaseConnection.TargetOffsetMode , BaseConnection.SourceOrientation , BaseConnection.TargetOrientation , BaseConnection.Direction , BaseConnection.DirectionalArrowsCount , BaseConnection.DirectionalArrowsOffset , BaseConnection.ArrowEnds , BaseConnection.ArrowShape , BaseConnection.Spacing , BaseConnection.ArrowSize , BaseConnection.SplitCommand , BaseConnection.DisconnectCommand , BaseConnection.Foreground , BaseConnection.Text , BaseConnection.FontSize , BaseConnection.FontFamily , BaseConnection.FontStyle , BaseConnection.FontWeight , BaseConnection.FontStretch , BaseConnection.DisconnectEvent , BaseConnection.SplitEvent , BaseConnection.Disconnect , BaseConnection.Split , BaseConnection.ZeroVector , BaseConnection.DefiningGeometry , BaseConnection.DrawDirectionalArrowheadGeometry(StreamGeometryContext, Vector, Point) , BaseConnection.DrawArrowGeometry(StreamGeometryContext, Point, Point, ConnectionDirection, ArrowHeadShape, Orientation) ,

BaseConnection.DrawDefaultArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.DrawRectangleArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.DrawEllipseArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.GetOffset() , BaseConnection.StartAnimation(double) , BaseConnection.StopAnimation() , BaseConnection.OnMouseDown(MouseButtonEventArgs) , BaseConnection.OnMouseUp(MouseButtonEventArgs) , BaseConnection.OnRender(DrawingContext)

Methods

DrawDirectionalArrowsGeometry(StreamGeometryContext, Point, Point)

```
protected override void DrawDirectionalArrowsGeometry(StreamGeometryContext context, Point source, Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

DrawLineGeometry(StreamGeometryContext, Point, Point)

```
protected override ((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource, Point ArrowEndTarget)) DrawLineGeometry(StreamGeometryContext context, Point source, Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

Returns

((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource, Point ArrowEndTarget))

GetTextPosition(FormattedText, Point, Point)

```
protected override Point GetTextPosition(FormattedText text, Point source, Point target)
```

Parameters

`text` FormattedText

`source` Point

`target` Point

Returns

Point

InterpolateCubicBezier(Point, Point, Point, Point, double)

```
protected static Point InterpolateCubicBezier(Point P0, Point P1, Point P2, Point P3,
double t)
```

Parameters

`P0` Point

`P1` Point

`P2` Point

`P3` Point

`t` double

Returns

Point

Enum ConnectionDirection

Namespace: Nodify

Assembly: Nodify.dll

The direction in which a connection is oriented.

```
public enum ConnectionDirection
```

Fields

`Backward = 1`

From Target to Source.

`Forward = 0`

From Source to Target.

Class ConnectionEventArgs

Namespace: Nodify

Assembly: Nodify.dll

Provides data for BaseConnection related routed events.

```
public class ConnectionEventArgs : RoutedEventArgs
```

Inheritance

object ← EventArgs ← RoutedEventArgs ← ConnectionEventArgs

Constructors

ConnectionEventArgs(object)

Initializes a new instance of the ConnectionEventArgs class using the specified Connection.

```
public ConnectionEventArgs(object connection)
```

Parameters

`connection` object

The DataContext of a related BaseConnection.

Properties

Connection

Gets the DataContext of the BaseConnection associated with this event.

```
public object Connection { get; }
```

Property Value

object

SplitLocation

Gets or sets the location where the connection should be split.

```
public Point SplitLocation { get; set; }
```

Property Value

Point

Methods

InvokeEventHandler(Delegate, object)

When overridden in a derived class, provides a way to invoke event handlers in a type-specific way, which can increase efficiency over the base implementation.

```
protected override void InvokeEventHandler(Delegate genericHandler, object genericTarget)
```

Parameters

`genericHandler` Delegate

The generic handler / delegate implementation to be invoked.

`genericTarget` object

The target on which the provided handler should be invoked.

Delegate ConnectionEventHandler

Namespace: Nodify

Assembly: Nodify.dll

Represents the method that will handle BaseConnection related routed events.

```
public delegate void ConnectionEventHandler(object sender, ConnectionEventArgs e)
```

Parameters

`sender` object

The object where the event handler is attached.

`e` ConnectionEventArgs

The event data.

Enum ConnectionOffsetMode

Namespace: Nodify

Assembly: Nodify.dll

Specifies the offset type that can be applied to a BaseConnection using the SourceOffset and the Target Offset values.

```
public enum ConnectionOffsetMode
```

Fields

`Circle = 1`

The offset is applied in a circle around the point.

`Edge = 3`

The offset is applied in a rectangle shape around the point, perpendicular to the edges.

`None = 0`

No offset applied.

`Rectangle = 2`

The offset is applied in a rectangle shape around the point.

`Static = 4`

The offset is applied as a fixed margin.

Class Connector

Namespace: Nodify

Assembly: Nodify.dll

Represents a connector control that can start and complete a PendingConnection. Has a Element Connector that the Anchor is calculated from for the PendingConnection. Center of this control is used if missing.

```
[TemplatePart(Name = "PART_Connector", Type = typeof(FrameworkElement))]
public class Connector : Control
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← Connector

Derived

NodeInput, NodeOutput, StateNode

Fields

AnchorProperty

```
public static readonly DependencyProperty AnchorProperty
```

Field Value

DependencyProperty

DisconnectCommandProperty

```
public static readonly DependencyProperty DisconnectCommandProperty
```

Field Value

DependencyProperty

DisconnectEvent

```
public static readonly RoutedEvent DisconnectEvent
```

Field Value

RoutedEvent

ElementConnector

```
protected const string ElementConnector = "PART_Connector"
```

Field Value

string

EnableOptimizations

Gets or sets if Connectors should enable optimizations based on OptimizeSafeZone and Optimize MinimumSelectedItems.

```
public static bool EnableOptimizations
```

Field Value

bool

IsConnectedProperty

```
public static readonly DependencyProperty IsConnectedProperty
```

Field Value

DependencyProperty

IsPendingConnectionProperty

```
public static readonly DependencyProperty IsPendingConnectionProperty
```

Field Value

DependencyProperty

OptimizeMinimumSelectedItems

Gets or sets the minimum selected items needed to trigger optimizations when outside of the Optimize SafeZone.

```
public static uint OptimizeMinimumSelectedItems
```

Field Value

uint

OptimizeSafeZone

Gets or sets the safe zone outside the editor's viewport that will not trigger optimizations.

```
public static double OptimizeSafeZone
```

Field Value

double

PendingConnectionCompletedEvent

```
public static readonly RoutedEvent PendingConnectionCompletedEvent
```

Field Value

RoutedEvent

PendingConnectionDragEvent

```
public static readonly RoutedEvent PendingConnectionDragEvent
```

Field Value

RoutedEvent

PendingConnectionStartedEvent

```
public static readonly RoutedEvent PendingConnectionStartedEvent
```

Field Value

RoutedEvent

Properties

AllowPendingConnectionCancellation

Gets or sets whether cancelling a pending connection is allowed.

```
public static bool AllowPendingConnectionCancellation { get; set; }
```

Property Value

bool

Anchor

Gets the location where Connections can be attached to. Bind with OneWayToSource

```
public Point Anchor { get; set; }
```

Property Value

Point

Container

Gets the ItemContainer that contains this Connector.

```
protected ItemContainer? Container { get; }
```

Property Value

ItemContainer

DisconnectCommand

Invoked if the Disconnect event is not handled. Parameter is the DataContext of this control.

```
public ICommand? DisconnectCommand { get; set; }
```

Property Value

ICommand

Editor

Gets the NodifyEditor that owns this Container.

```
protected NodifyEditor? Editor { get; }
```

Property Value

NodifyEditor

EnableStickyConnections

Gets or sets whether the connection should be completed in two steps.

```
public static bool EnableStickyConnections { get; set; }
```

Property Value

bool

IsConnected

If this is set to false, the Disconnect event will not be invoked and the connector will stop updating its Anchor when moved, resized etc.

```
public bool IsConnected { get; set; }
```

Property Value

bool

IsPendingConnection

Gets a value that indicates whether a PendingConnection is in progress for this Connector.

```
public bool IsPendingConnection { get; protected set; }
```

Property Value

bool

Thumb

Gets the FrameworkElement used to calculate the Anchor.

```
protected FrameworkElement? Thumb { get; }
```

Property Value

FrameworkElement

Methods

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

OnConnectorDrag(Vector)

```
protected virtual void OnConnectorDrag(Vector offset)
```

Parameters

`offset` Vector

OnConnectorDragCompleted(bool)

```
protected virtual void OnConnectorDragCompleted(bool cancel = false)
```

Parameters

`cancel` bool

OnConnectorDragStarted()

```
protected virtual void OnConnectorDragStarted()
```

OnDisconnect()

```
protected virtual void OnDisconnect()
```

OnKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

OnLostMouseCapture(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.LostMouseCapture attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnLostMouseCapture(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains event data.

OnMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

OnMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

OnMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

OnRenderSizeChanged(SizeChangedInfo)

Raises the SizeChanged event, using the specified information as part of the eventual event data.

```
protected override void OnRenderSizeChanged(SizeChangedInfo sizeInfo)
```

Parameters

`sizeInfo` SizeChangedInfo

Details of the old and new size involved in the change.

UpdateAnchor()

Updates the Anchor based on Container's location.

```
public void UpdateAnchor()
```

UpdateAnchor(Point)

Updates the Anchor relative to a location. (usually Container's location)

```
protected void UpdateAnchor(Point location)
```

Parameters

`location` Point

The relative location

UpdateAnchorOptimized(Point)

Updates the Anchor and applies optimizations if needed based on EnableOptimizations flag

```
protected void UpdateAnchorOptimized(Point location)
```

Parameters

`location` Point

Events

Disconnect

Triggered by the EditorGestures.Connector.Disconnect gesture.

```
public event ConnectorEventHandler Disconnect
```

Event Type

ConnectorEventHandler

PendingConnectionCompleted

Triggered by the EditorGestures.Connector.Connect gesture.

```
public event PendingConnectionEventHandler PendingConnectionCompleted
```

Event Type

PendingConnectionEventHandler

PendingConnectionDrag

Occurs when the mouse is changing position and the Connector has mouse capture.

```
public event PendingConnectionEventHandler PendingConnectionDrag
```

Event Type

PendingConnectionEventHandler

PendingConnectionStarted

Triggered by the EditorGestures.Connector.Connect gesture.

```
public event PendingConnectionEventHandler PendingConnectionStarted
```

Event Type

PendingConnectionEventHandler

Class ConnectorEventArgs

Namespace: Nodify

Assembly: Nodify.dll

Provides data for Connector related routed events.

```
public class ConnectorEventArgs : RoutedEventArgs
```

Inheritance

object ← EventArgs ← RoutedEventArgs ← ConnectorEventArgs

Constructors

ConnectorEventArgs(object)

Initializes a new instance of the ConnectorEventArgs class using the specified Connector.

```
public ConnectorEventArgs(object connector)
```

Parameters

`connector` object

The DataContext of a related Connector.

Properties

Anchor

Gets or sets the Anchor of the Connector associated with this event.

```
public Point Anchor { get; set; }
```

Property Value

Point

Connector

Gets the DataContext of the Connector associated with this event.

```
public object Connector { get; }
```

Property Value

object

Methods

InvokeEventHandler(Delegate, object)

When overridden in a derived class, provides a way to invoke event handlers in a type-specific way, which can increase efficiency over the base implementation.

```
protected override void InvokeEventHandler(Delegate genericHandler, object genericTarget)
```

Parameters

`genericHandler` Delegate

The generic handler / delegate implementation to be invoked.

`genericTarget` object

The target on which the provided handler should be invoked.

Delegate ConnectorEventHandler

Namespace: Nodify

Assembly: Nodify.dll

Represents the method that will handle Connector related routed events.

```
public delegate void ConnectorEventHandler(object sender, ConnectorEventArgs e)
```

Parameters

`sender` object

The object where the event handler is attached.

`e` ConnectorEventArgs

The event data.

Class ContainerDefaultState

Namespace: Nodify

Assembly: Nodify.dll

The default state of the ItemContainer.

```
public class ContainerDefaultState : ContainerState
```

Inheritance

object ← ContainerState ← ContainerDefaultState

Inherited Members

ContainerState.Container , ContainerState.Editor , ContainerState.HandleMouseWheel(MouseWheelEventArgs) , ContainerState.HandleKeyUp(KeyEventArgs) , ContainerState.HandleKeyDown(KeyEventArgs) , ContainerState.Enter(ContainerState) , ContainerState.Exit() , ContainerState.PushState(ContainerState) , ContainerState.PopState()

Constructors

ContainerDefaultState(ItemContainer)

Creates a new instance of the ContainerDefaultState.

```
public ContainerDefaultState(ItemContainer container)
```

Parameters

`container` ItemContainer

The owner of the state.

Methods

HandleMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

ReEnter(ContainerState)

Called when PopState() is called.

```
public override void ReEnter(ContainerState from)
```

Parameters

`from` ContainerState

The state we re-enter from.

Class ContainerDraggingState

Namespace: Nodify

Assembly: Nodify.dll

Dragging state of the container.

```
public class ContainerDraggingState : ContainerState
```

Inheritance

object ← ContainerState ← ContainerDraggingState

Inherited Members

ContainerState.Container , ContainerState.Editor , ContainerState.HandleMouseDown(MouseButtonEventArgs) , ContainerState.HandleMouseWheel(MouseWheelEventArgs) , ContainerState.HandleKeyDown(KeyEventArgs) , ContainerState.ReEnter(ContainerState) , ContainerState.PushState(ContainerState) , ContainerState.PopState()

Constructors

ContainerDraggingState(ItemContainer)

Constructs an instance of the ContainerDraggingState state.

```
public ContainerDraggingState(ItemContainer container)
```

Parameters

`container` ItemContainer

The owner of the state.

Properties

Canceled

```
public bool Canceled { get; set; }
```

Property Value

bool

Methods

Enter(ContainerState?)

Called when PushState(ContainerState) or PopState() is called.

```
public override void Enter(ContainerState? from)
```

Parameters

`from` ContainerState

The state we enter from (is null for root state).

Exit()

Called when PopState() is called.

```
public override void Exit()
```

HandleKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

Class ContainerState

Namespace: Nodify

Assembly: Nodify.dll

The base class for container states.

```
public abstract class ContainerState
```

Inheritance

object ← ContainerState

Derived

ContainerDefaultState, ContainerDraggingState

Constructors

ContainerState(ItemContainer)

Constructs a new ContainerState.

```
public ContainerState(ItemContainer container)
```

Parameters

`container` ItemContainer

The owner of the state.

Properties

Container

The owner of the state.

```
protected ItemContainer Container { get; }
```

Property Value

ItemContainer

Editor

The owner of the state.

```
protected NodifyEditor Editor { get; }
```

Property Value

NodifyEditor

Methods

Enter(ContainerState?)

Called when PushState(ContainerState) or PopState() is called.

```
public virtual void Enter(ContainerState? from)
```

Parameters

`from` ContainerState

The state we enter from (is null for root state).

Exit()

Called when PopState() is called.

```
public virtual void Exit()
```

HandleKeyDown(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleKeyDown(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

HandleKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

HandleMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

HandleMouseWheel(MouseWheelEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseWheel attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseWheel(MouseWheelEventArgs e)
```

Parameters

`e` MouseWheelEventArgs

The MouseWheelEventArgs that contains the event data.

PopState()

Pops the current state from the stack.

```
public virtual void PopState()
```

PushState(ContainerState)

Pushes a new state into the stack.

```
public virtual void PushState(ContainerState newState)
```

Parameters

`newState` ContainerState

The new state.

ReEnter(ContainerState)

Called when PopState() is called.

```
public virtual void ReEnter(ContainerState from)
```

Parameters

`from` ContainerState

The state we re-enter from.

Class DecoratorContainer

Namespace: Nodify

Assembly: Nodify.dll

The container for all the items generated from the Decorators collection.

```
public class DecoratorContainer : ContentControl, INodifyCanvasItem
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← DecoratorContainer

Implements

INodifyCanvasItem

Fields

ActualSizeProperty

```
public static readonly DependencyProperty ActualSizeProperty
```

Field Value

DependencyProperty

LocationChangedEvent

```
public static readonly RoutedEvent LocationChangedEvent
```

Field Value

RoutedEvent

LocationProperty

```
public static readonly DependencyProperty LocationProperty
```

Field Value

DependencyProperty

Properties

ActualSize

Gets the actual size of this DecoratorContainer.

```
public Size ActualSize { get; set; }
```

Property Value

Size

Location

Gets or sets the location of this DecoratorContainer inside the NodifyEditor.DecoratorsHost.

```
public Point Location { get; set; }
```

Property Value

Point

Methods

OnLocationChanged()

Raises the LocationChangedEvent.

```
protected void OnLocationChanged()
```

OnRenderSizeChanged(SizeChangedInfo)

Raises the SizeChanged event, using the specified information as part of the eventual event data.

```
protected override void OnRenderSizeChanged(SizeChangedInfo sizeInfo)
```

Parameters

`sizeInfo` SizeChangedInfo

Details of the old and new size involved in the change.

Events

LocationChanged

Occurs when the Location of this DecoratorContainer is changed.

```
public event RoutedEventHandler LocationChanged
```

Event Type

RoutedEventHandler

Class EditorCommands

Namespace: Nodify

Assembly: Nodify.dll

```
public static class EditorCommands
```

Inheritance

object ← EditorCommands

Properties

Align

Aligns SelectedItems using the specified alignment method. Parameter is of type EditorCommands.Alignment or a string that can be converted to an alignment.

```
public static RoutedUICommand Align { get; }
```

Property Value

RoutedUICommand

BringIntoView

Moves the ViewportLocation to the specified location. Parameter is a Point or a string that can be converted to a point.

```
public static RoutedUICommand BringIntoView { get; }
```

Property Value

RoutedUICommand

FitToScreen

Scales the editor's viewport to fit all the ItemContainers if that's possible.

```
public static RoutedUICommand FitToScreen { get; }
```

Property Value

RoutedUICommand

SelectAll

Select all ItemContainers in the NodifyEditor.

```
public static RoutedUICommand SelectAll { get; }
```

Property Value

RoutedUICommand

ZoomIn

Zoom in relative to the editor's viewport center.

```
public static RoutedUICommand ZoomIn { get; }
```

Property Value

RoutedUICommand

ZoomOut

Zoom out relative to the editor's viewport center.

```
public static RoutedUICommand ZoomOut { get; }
```

Property Value

RoutedUICommand

Enum EditorCommands.Alignment

Namespace: Nodify

Assembly: Nodify.dll

Specifies the possible alignment values used by the Align command.

```
public enum EditorCommands.Alignment
```

Fields

`Bottom = 2`

`Center = 5`

`Left = 1`

`Middle = 4`

`Right = 3`

`Top = 0`

Class EditorDefaultState

Namespace: Nodify

Assembly: Nodify.dll

The default state of the editor.

Default State
- mouse left down -> Selecting State
- mouse right down -> Panning State

Selecting State
- mouse left up -> Default State
- mouse right down -> Panning State

Panning State
- mouse right up -> previous state (Selecting State or Default State)
- mouse left up -> Panning State

```
public class EditorDefaultState : EditorState
```

Inheritance

object ← EditorState ← EditorDefaultState

Inherited Members

EditorState.Editor , EditorState.HandleMouseUp(MouseButtonEventArgs) , EditorState.HandleMouseMove(MouseEventArgs) , EditorState.HandleMouseWheel(MouseWheelEventArgs) , EditorState.HandleAutoPanning(MouseEventArgs) , EditorState.HandleKeyUp(KeyEventArgs) , EditorState.HandleKeyDown(KeyEventArgs) , EditorState.Enter(EditorState) , EditorState.Exit() , EditorState.ReEnter(EditorState) , EditorState.PushState(EditorState) , EditorState.PopState()

Constructors

EditorDefaultState(NodifyEditor)

Constructs an instance of the EditorDefaultState state.

```
public EditorDefaultState(NodifyEditor editor)
```

Parameters

`editor` NodifyEditor

The owner of the state.

Methods

HandleMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseDown(MouseButtonEventArgs e)
```

Parameters

`e` MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

Class EditorGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures used by built-in controls inside the NodifyEditor.

```
public class EditorGestures
```

Inheritance

object ← EditorGestures

Fields

Mappings

```
public static readonly EditorGestures Mappings
```

Field Value

EditorGestures

Properties

Connection

Gestures for the connection.

```
public EditorGestures.ConnectionGestures Connection { get; }
```

Property Value

EditorGestures.ConnectionGestures

Connector

Gestures for the connector.

```
public EditorGestures.ConnectorGestures Connector { get; }
```

Property Value

EditorGestures.ConnectorGestures

Editor

Gestures for the editor.

```
public EditorGestures.NodifyEditorGestures Editor { get; }
```

Property Value

EditorGestures.NodifyEditorGestures

GroupingNode

Gestures for the grouping node.

```
public EditorGestures.GroupingNodeGestures GroupingNode { get; }
```

Property Value

EditorGestures.GroupingNodeGestures

ItemContainer

Gestures for the item container.

```
public EditorGestures.ItemContainerGestures ItemContainer { get; }
```

Property Value

EditorGestures.ItemContainerGestures

Methods

Apply(EditorGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures gestures)
```

Parameters

`gestures` EditorGestures

The gestures to copy.

Class EditorGestures.ConnectionGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures used by the BaseConnection.

```
public class EditorGestures.ConnectionGestures
```

Inheritance

object ← EditorGestures.ConnectionGestures

Constructors

ConnectionGestures()

```
public ConnectionGestures()
```

Properties

Disconnect

Gesture to call the DisconnectCommand command.

```
public InputGestureRef Disconnect { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Alt+LeftClick.

Split

Gesture to call the SplitCommand command.

```
public InputGestureRef Split { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to LeftDoubleClick.

Methods

Apply(ConnectionGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.ConnectionGestures gestures)
```

Parameters

`gestures` EditorGestures.ConnectionGestures

The gestures to copy.

Class EditorGestures.ConnectorGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures used by the Connector.

```
public class EditorGestures.ConnectorGestures
```

Inheritance

object ← EditorGestures.ConnectorGestures

Constructors

ConnectorGestures()

```
public ConnectorGestures()
```

Properties

CancelAction

Gesture to cancel the pending connection.

```
public InputGestureRef CancelAction { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to RightClick or Escape.

Connect

Gesture to start and complete a pending connection.

```
public InputGestureRef Connect { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to LeftClick.

Disconnect

Gesture to call the DisconnectCommand.

```
public InputGestureRef Disconnect { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Alt+LeftClick.

Methods

Apply(ConnectorGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.ConnectorGestures gestures)
```

Parameters

`gestures` EditorGestures.ConnectorGestures

The gestures to copy.

Class EditorGestures.GroupingNodeGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures for the GroupingNode.

```
public class EditorGestures.GroupingNodeGestures
```

Inheritance

object ← EditorGestures.GroupingNodeGestures

Properties

SwitchMovementMode

The key modifier that will toggle between GroupingMovementModes.

```
public ModifierKeys SwitchMovementMode { get; set; }
```

Property Value

ModifierKeys

Remarks

The modifier must be allowed by the ItemContainer.Drag gesture.
Defaults to Shift.

Methods

Apply(GroupingNodeGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.GroupingNodeGestures gestures)
```

Parameters

`gestures` EditorGestures.GroupingNodeGestures

The gestures to copy.

Class EditorGestures.ItemContainerGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures for the item containers.

```
public class EditorGestures.ItemContainerGestures
```

Inheritance

object ← EditorGestures.ItemContainerGestures

Constructors

ItemContainerGestures()

```
public ItemContainerGestures()
```

Properties

CancelAction

Gesture to cancel the dragging operation.

```
public InputGestureRef CancelAction { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to RightClick or Escape.

Drag

Gesture to start and complete a dragging operation.

```
public InputGestureRef Drag { get; }
```

Property Value

InputGestureRef

Remarks

Using a Selection strategy to drag from a new selection.
Defaults to any of the Selection gestures.

Selection

Gesture to select the container using a EditorGestures.SelectionGestures strategy.

```
public EditorGestures.SelectionGestures Selection { get; }
```

Property Value

EditorGestures.SelectionGestures

Remarks

Defaults to RightClick or any of the EditorGestures.SelectionGestures gestures.

Methods

Apply(ItemContainerGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.ItemContainerGestures gestures)
```

Parameters

`gestures` EditorGestures.ItemContainerGestures

The gestures to copy.

Class EditorGestures.NodifyEditorGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures for the editor.

```
public class EditorGestures.NodifyEditorGestures
```

Inheritance

object ← EditorGestures.NodifyEditorGestures

Constructors

NodifyEditorGestures()

```
public NodifyEditorGestures()
```

Properties

FitToScreen

Gesture used to fit as many containers as possible into the viewport.

```
public InputGestureRef FitToScreen { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Shift+Home.

Pan

Gesture used to start panning.

```
public InputGestureRef Pan { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to RightClick or MiddleClick.

ResetViewportLocation

Gesture used to move the editor's viewport location to (0, 0).

```
public InputGestureRef ResetViewportLocation { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Home.

Selection

Gesture used to start selecting using a EditorGestures.SelectionGestures strategy.

```
public EditorGestures.SelectionGestures Selection { get; }
```

Property Value

EditorGestures.SelectionGestures

ZoomIn

Gesture used to zoom in.

```
public InputGestureRef ZoomIn { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Control+OemPlus.

ZoomModifierKey

The key modifier required to start zooming by mouse wheel.

```
public ModifierKeys ZoomModifierKey { get; set; }
```

Property Value

ModifierKeys

Remarks

Defaults to None.

ZoomOut

Gesture used to zoom out.

```
public InputGestureRef ZoomOut { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Control+OemMinus.

Methods

Apply(NodifyEditorGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.NodifyEditorGestures gestures)
```

Parameters

`gestures` EditorGestures.NodifyEditorGestures

The gestures to copy.

Class EditorGestures.SelectionGestures

Namespace: Nodify

Assembly: Nodify.dll

Gestures for the selection.

```
public class EditorGestures.SelectionGestures
```

Inheritance

object ← EditorGestures.SelectionGestures

Constructors

SelectionGestures()

```
public SelectionGestures()
```

SelectionGestures(MouseAction)

```
public SelectionGestures(MouseAction mouseAction)
```

Parameters

`mouseAction` MouseAction

Fields

None

Disable selection gestures.

```
public static readonly EditorGestures.SelectionGestures None
```

Field Value

EditorGestures.SelectionGestures

Properties

Append

Gesture to add the new selected items to the previous selection.

```
public InputGestureRef Append { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Shift+LeftClick.

Cancel

Cancel the current selection operation reverting to the previous selection.

```
public InputGestureRef Cancel { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Escape.

Invert

Gesture to invert the selected items.

```
public InputGestureRef Invert { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Control+LeftClick.

Remove

Gesture to remove the selected items from the previous selection.

```
public InputGestureRef Remove { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to Alt+LeftClick.

Replace

Gesture to replace previous selection with the selected items.

```
public InputGestureRef Replace { get; }
```

Property Value

InputGestureRef

Remarks

Defaults to LeftClick.

Select

Gesture used to start selecting using a EditorGestures.SelectionGestures strategy.

```
public InputGestureRef Select { get; }
```

Property Value

InputGestureRef

Methods

Apply(SelectionGestures)

Copies from the specified gestures.

```
public void Apply(EditorGestures.SelectionGestures gestures)
```

Parameters

`gestures` EditorGestures.SelectionGestures

The gestures to copy.

Class EditorPanningState

Namespace: Nodify

Assembly: Nodify.dll

The panning state of the editor.

```
public class EditorPanningState : EditorState
```

Inheritance

object ← EditorState ← EditorPanningState

Inherited Members

EditorState.Editor , EditorState.HandleMouseDown(MouseButtonEventArgs) , EditorState.HandleMouseWheel(MouseWheelEventArgs) , EditorState.HandleAutoPanning(MouseEventArgs) , EditorState.HandleKeyUp(KeyEventArgs) , EditorState.HandleKeyDown(KeyEventArgs) , EditorState.ReEnter(EditorState) , EditorState.PushState(EditorState) , EditorState.PopState()

Constructors

EditorPanningState(NodifyEditor)

Constructs an instance of the EditorPanningState state.

```
public EditorPanningState(NodifyEditor editor)
```

Parameters

`editor` NodifyEditor

The owner of the state.

Methods

Enter(EditorState?)

Called when PushState(EditorState) is called.

```
public override void Enter(EditorState? from)
```

Parameters

`from` EditorState

The state we enter from (is null for root state).

Exit()

Called when PopState() is called.

```
public override void Exit()
```

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseMove(MouseEventArgs e)
```

Parameters

`e` MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

Class EditorSelectingState

Namespace: Nodify

Assembly: Nodify.dll

The selecting state of the editor.

```
public class EditorSelectingState : EditorState
```

Inheritance

object ← EditorState ← EditorSelectingState

Inherited Members

EditorState.Editor , EditorState.HandleMouseWheel(MouseWheelEventArgs) , EditorState.HandleKeyDown(KeyEventArgs) , EditorState.ReEnter(EditorState) , EditorState.PushState(EditorState) , EditorState.PopState()

Constructors

EditorSelectingState(NodifyEditor, SelectionType)

Constructs an instance of the EditorSelectingState state.

```
public EditorSelectingState(NodifyEditor editor, SelectionHelper.SelectionType type)
```

Parameters

`editor` NodifyEditor

The owner of the state.

`type` SelectionHelper.SelectionType

Properties

Selection

The selection helper.

```
protected SelectionHelper Selection { get; }
```

Property Value

SelectionHelper

Methods

Enter(EditorState?)

Called when PushState(EditorState) is called.

```
public override void Enter(EditorState? from)
```

Parameters

`from` EditorState

The state we enter from (is null for root state).

Exit()

Called when PopState() is called.

```
public override void Exit()
```

HandleAutoPanning(MouseEventArgs)

Handles auto panning when mouse is outside the editor.

```
public override void HandleAutoPanning(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

HandleMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public override void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

Class EditorState

Namespace: Nodify

Assembly: Nodify.dll

The base class for editor states.

```
public abstract class EditorState
```

Inheritance

object ← EditorState

Derived

EditorDefaultState, EditorPanningState, EditorSelectingState

Constructors

EditorState(NodifyEditor)

Constructs a new EditorState.

```
public EditorState(NodifyEditor editor)
```

Parameters

`editor` NodifyEditor

The owner of the state.

Properties

Editor

The owner of the state.

```
protected NodifyEditor Editor { get; }
```

Property Value

NodifyEditor

Methods

Enter(EditorState?)

Called when PushState(EditorState) is called.

```
public virtual void Enter(EditorState? from)
```

Parameters

`from` EditorState

The state we enter from (is null for root state).

Exit()

Called when PopState() is called.

```
public virtual void Exit()
```

HandleAutoPanning(MouseEventArgs)

Handles auto panning when mouse is outside the editor.

```
public virtual void HandleAutoPanning(MouseEventArgs e)
```

Parameters

`e` MouseEventArgs

The MouseEventArgs that contains the event data.

HandleKeyDown(KeyEventArgs)

Invoked when the KeyDown event is received.

```
public virtual void HandleKeyDown(KeyEventArgs e)
```

Parameters

e KeyEventArgs

Information about the event.

HandleKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

HandleMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

HandleMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

HandleMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

HandleMouseWheel(MouseWheelEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseWheel attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
public virtual void HandleMouseWheel(MouseWheelEventArgs e)
```

Parameters

`e` MouseWheelEventArgs

The MouseWheelEventArgs that contains the event data.

PopState()

Pops the current state from the stack.

```
public virtual void PopState()
```

PushState(EditorState)

Pushes a new state into the stack.

```
public virtual void PushState(EditorState newState)
```

Parameters

`newState` EditorState

The new state.

ReEnter(EditorState)

Called when PopState() is called.

```
public virtual void ReEnter(EditorState from)
```

Parameters

`from` EditorState

The state we re-enter from.

Enum GroupingMovementMode

Namespace: Nodify

Assembly: Nodify.dll

Specifies the possible movement modes of a GroupingNode.

```
public enum GroupingMovementMode
```

Fields

`Group = 0`

The GroupingNode will move its content when moved.

`Self = 1`

The GroupingNode will not move its content when moved.

Class GroupingNode

Namespace: Nodify

Assembly: Nodify.dll

Defines a panel with a header that groups ItemContainers inside it and can be resized.

```
[TemplatePart(Name = "PART_ResizeThumb", Type = typeof(FrameworkElement))]
[TemplatePart(Name = "PART_Header", Type = typeof(FrameworkElement))]
[TemplatePart(Name = "PART_Content", Type = typeof(FrameworkElement))]
public class GroupingNode : HeaderedContentControl
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← HeaderedContentControl ← GroupingNode

Constructors

GroupingNode()

Initializes a new instance of the GroupingNode class.

```
public GroupingNode()
```

Fields

ActualSizeProperty

```
public static readonly DependencyProperty ActualSizeProperty
```

Field Value

DependencyProperty

CanResizeProperty

```
public static readonly DependencyProperty CanResizeProperty
```

Field Value

DependencyProperty

ContentControl

Gets the ContentControl control of this GroupingNode.

```
protected FrameworkElement? ContentControl
```

Field Value

FrameworkElement

ElementContent

```
protected const string ElementContent = "PART_Content"
```

Field Value

string

ElementHeader

```
protected const string ElementHeader = "PART_Header"
```

Field Value

string

ElementResizeThumb

```
protected const string ElementResizeThumb = "PART_ResizeThumb"
```

Field Value

string

GroupMovementBoxed

```
protected static readonly object GroupMovementBoxed
```

Field Value

object

HeaderBrushProperty

```
public static readonly DependencyProperty HeaderBrushProperty
```

Field Value

DependencyProperty

HeaderControl

Gets the Header control of this GroupingNode.

```
protected FrameworkElement? HeaderControl
```

Field Value

FrameworkElement

MovementModeProperty

```
public static readonly DependencyProperty MovementModeProperty
```

Field Value

DependencyProperty

ResizeCompletedCommandProperty

```
public static readonly DependencyProperty ResizeCompletedCommandProperty
```

Field Value

DependencyProperty

ResizeCompletedEvent

```
public static readonly RoutedEvent ResizeCompletedEvent
```

Field Value

RoutedEvent

ResizeStartedCommandProperty

```
public static readonly DependencyProperty ResizeStartedCommandProperty
```

Field Value

DependencyProperty

ResizeStartedEvent

```
public static readonly RoutedEvent ResizeStartedEvent
```

Field Value

RoutedEvent

ResizeThumb

Gets the FrameworkElement used to resize this GroupingNode.

```
protected FrameworkElement? ResizeThumb
```

Field Value

FrameworkElement

Properties

ActualSize

Gets or sets the actual size of this GroupingNode.

```
public Size ActualSize { get; set; }
```

Property Value

Size

CanResize

Gets or sets a value that indicates whether this GroupingNode can be resized.

```
public bool CanResize { get; set; }
```

Property Value

bool

Container

Gets the NodifyEditor that owns this Container.

```
protected ItemContainer? Container { get; }
```

Property Value

ItemContainer

Editor

Gets the NodifyEditor that owns this GroupingNode.

```
protected NodifyEditor? Editor { get; }
```

Property Value

NodifyEditor

HeaderBrush

Gets or sets the brush used for the background of the Header of this GroupingNode.

```
public Brush HeaderBrush { get; set; }
```

Property Value

Brush

MovementMode

Gets or sets the default movement mode which can be temporarily changed by holding the SwitchMovementModeModifierKey while dragging by the header.

```
public GroupingMovementMode MovementMode { get; set; }
```

Property Value

GroupingMovementMode

ResizeCompletedCommand

Invoked when the ResizeCompleted event is not handled. Parameter is the ActualSize of the container.

```
public ICommand? ResizeCompletedCommand { get; set; }
```

Property Value

ICommand

ResizeStartedCommand

Invoked when the ResizeStarted event is not handled. Parameter is the ActualSize of the container.

```
public ICommand? ResizeStartedCommand { get; set; }
```

Property Value

ICommand

Methods

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

Events

ResizeCompleted

Occurs when the node finished resizing.

```
public event ResizeEventHandler ResizeCompleted
```

Event Type

ResizeEventHandler

ResizeStarted

Occurs when the node started resizing.

```
public event ResizeEventHandler ResizeStarted
```

Event Type

ResizeEventHandler

Interface INodifyCanvasItem

Namespace: Nodify

Assembly: Nodify.dll

Interface for items inside a NodifyCanvas.

```
public interface INodifyCanvasItem
```

Properties

DesiredSize

The desired size of the item.

```
Size DesiredSize { get; }
```

Property Value

Size

Location

The location of the item.

```
Point Location { get; }
```

Property Value

Point

Methods

Arrange(Rect)

Positions child elements and determines a size for a UIElement. Parent elements call this method from their ArrangeCore(Rect) implementation (or a WPF framework-level equivalent) to form a recursive layout update. This method constitutes the second pass of a layout update.

```
void Arrange(Rect rect)
```

Parameters

`rect` Rect

Class InputGestureRef

Namespace: Nodify

Assembly: Nodify.dll

An input gesture that allows changing its logic at runtime without changing its reference. Useful for classes that capture the object reference without the posibility of updating it. (e.g. EditorCommands)

```
public sealed class InputGestureRef : InputGesture
```

Inheritance

object ← InputGesture ← InputGestureRef

Properties

Value

The referenced gesture.

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

Property Value

InputGesture

Methods

Matches(object, InputEventArgs)

When overridden in a derived class, determines whether the specified InputGesture matches the input associated with the specified InputEventArgs object.

```
public override bool Matches(object targetElement, InputEventArgs inputEventArgs)
```

Parameters

`targetElement` object

The target of the command.

`inputEventArgs` InputEventArgs

The input event data to compare this gesture to.

Returns

bool

true if the gesture matches the input; otherwise, false.

Operators

implicit operator InputGestureRef(MultiGesture)

```
public static implicit operator InputGestureRef(MultiGesture gesture)
```

Parameters

`gesture` MultiGesture

Returns

InputGestureRef

implicit operator InputGestureRef(KeyGesture)

```
public static implicit operator InputGestureRef(KeyGesture gesture)
```

Parameters

`gesture` KeyGesture

Returns

InputGestureRef

implicit operator InputGestureRef(MouseGesture)

```
public static implicit operator InputGestureRef(MouseGesture gesture)
```

Parameters

`gesture` MouseGesture

Returns

InputGestureRef

Class ItemContainer

Namespace: Nodify

Assembly: Nodify.dll

The container for all the items generated by the ItemsSource of the NodifyEditor.

```
public class ItemContainer : ContentControl, INodifyCanvasItem
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← ItemContainer

Implements

INodifyCanvasItem

Constructors

ItemContainer(NodifyEditor)

Constructs an instance of an ItemContainer in the specified NodifyEditor.

```
public ItemContainer(NodifyEditor editor)
```

Parameters

`editor` NodifyEditor

Fields

ActualSizeProperty

```
public static readonly DependencyProperty ActualSizeProperty
```

Field Value

DependencyProperty

DesiredSizeForSelectionProperty

```
public static readonly DependencyProperty DesiredSizeForSelectionProperty
```

Field Value

DependencyProperty

DragCompletedEvent

```
public static readonly RoutedEvent DragCompletedEvent
```

Field Value

RoutedEvent

DragDeltaEvent

```
public static readonly RoutedEvent DragDeltaEvent
```

Field Value

RoutedEvent

DragStartedEvent

```
public static readonly RoutedEvent DragStartedEvent
```

Field Value

RoutedEvent

HighlightBrushProperty

```
public static readonly DependencyProperty HighlightBrushProperty
```

Field Value

DependencyProperty

IsDraggableProperty

```
public static readonly DependencyProperty IsDraggableProperty
```

Field Value

DependencyProperty

IsPreviewingLocationProperty

```
public static readonly DependencyProperty IsPreviewingLocationProperty
```

Field Value

DependencyProperty

IsPreviewingLocationPropertyKey

```
public static readonly DependencyPropertyKey IsPreviewingLocationPropertyKey
```

Field Value

DependencyPropertyKey

IsPreviewingSelectionProperty

```
public static readonly DependencyProperty IsPreviewingSelectionProperty
```

Field Value

DependencyProperty

IsPreviewingSelectionPropertyKey

```
public static readonly DependencyPropertyKey IsPreviewingSelectionPropertyKey
```

Field Value

DependencyPropertyKey

IsSelectableProperty

```
public static readonly DependencyProperty IsSelectableProperty
```

Field Value

DependencyProperty

IsSelectedProperty

```
public static readonly DependencyProperty IsSelectedProperty
```

Field Value

DependencyProperty

LocationChangedEvent

```
public static readonly RoutedEvent LocationChangedEvent
```

Field Value

RoutedEvent

LocationProperty

```
public static readonly DependencyProperty LocationProperty
```

Field Value

DependencyProperty

SelectedBorderThicknessProperty

```
public static readonly DependencyProperty SelectedBorderThicknessProperty
```

Field Value

DependencyProperty

SelectedBrushProperty

```
public static readonly DependencyProperty SelectedBrushProperty
```

Field Value

DependencyProperty

SelectedEvent

```
public static readonly RoutedEvent SelectedEvent
```

Field Value

RoutedEvent

UnselectedEvent

```
public static readonly RoutedEvent UnselectedEvent
```

Field Value

RoutedEvent

Properties

ActualSize

Gets the actual size of this ItemContainer.

```
public Size ActualSize { get; set; }
```

Property Value

Size

AllowDraggingCancellation

Gets or sets whether cancelling a dragging operation is allowed.

```
public static bool AllowDraggingCancellation { get; set; }
```

Property Value

bool

DesiredSizeForSelection

Overrides the size to check against when calculating if this ItemContainer can be part of the current SelectedArea. Defaults to RenderSize.

```
public Size? DesiredSizeForSelection { get; set; }
```

Property Value

Size?

Editor

The NodifyEditor that owns this ItemContainer.

```
public NodifyEditor Editor { get; }
```

Property Value

NodifyEditor

HighlightBrush

Gets or sets the brush used when the IsOverElementProperty attached property is true for this ItemContainer.

```
public Brush HighlightBrush { get; set; }
```

Property Value

Brush

IsDraggable

Gets or sets whether this ItemContainer can be dragged.

```
public bool IsDraggable { get; set; }
```

Property Value

bool

IsPreviewingLocation

Gets a value indicating whether this ItemContainer is previewing a new location but didn't logically move there.

```
public bool IsPreviewingLocation { get; protected set; }
```

Property Value

bool

IsPreviewingSelection

Gets a value indicating whether this ItemContainer is about to change its IsSelected state.

```
public bool? IsPreviewingSelection { get; }
```

Property Value

bool?

IsSelectable

Gets or sets whether this ItemContainer can be selected.

```
public bool IsSelectable { get; set; }
```

Property Value

bool

IsSelected

Gets or sets a value that indicates whether this ItemContainer is selected. Can only be set if IsSelectable is true.

```
public bool IsSelected { get; set; }
```

Property Value

bool

Location

Gets or sets the location of this ItemContainer inside the NodifyEditor in graph space coordinates.

```
public Point Location { get; set; }
```

Property Value

Point

SelectedBorderThickness

Gets or sets the border thickness used when IsSelected or IsPreviewingSelection is true.

```
public Thickness SelectedBorderThickness { get; set; }
```

Property Value

Thickness

SelectedBrush

Gets or sets the brush used when IsSelected or IsPreviewingSelection is true.

```
public Brush SelectedBrush { get; set; }
```

Property Value

Brush

SelectedMargin

The calculated margin when the container is selected or previewing selection.

```
public Thickness SelectedMargin { get; }
```

Property Value

Thickness

State

The current state of the container.

```
public ContainerState State { get; }
```

Property Value

ContainerState

Methods

GetInitialState()

Creates the initial state of the container.

```
protected virtual ContainerState GetInitialState()
```

Returns

ContainerState

The initial state.

IsSelectableInArea(Rect, bool)

Checks if `area` contains or intersects with this ItemContainer taking into consideration the DesiredSizeForSelection.

```
public virtual bool IsSelectableInArea(Rect area, bool isContained)
```

Parameters

`area` Rect

The area to check if contains or intersects this ItemContainer.

`isContained` bool

If true will check if `area` contains this, otherwise will check if `area` intersects with this.

Returns

bool

True if `area` contains or intersects this ItemContainer.

IsSelectableLocation(Point)

Checks if `position` is selectable.

```
protected virtual bool IsSelectableLocation(Point position)
```

Parameters

`position` Point

A position relative to this ItemContainer.

Returns

bool

True if `position` is selectable.

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

OnKeyDown(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnKeyDown(KeyEventArgs e)
```

Parameters

`e` KeyEventArgs

The KeyEventArgs that contains the event data.

OnKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

OnLocationChanged()

Raises the LocationChangedEvent and sets IsPreviewingLocation to false.

```
protected void OnLocationChanged()
```

OnLostMouseCapture(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.LostMouseCapture attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnLostMouseCapture(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains event data.

OnMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

OnMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

OnMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

OnMouseWheel(MouseWheelEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseWheel attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseWheel(MouseWheelEventArgs e)
```

Parameters

`e` MouseWheelEventArgs

The MouseWheelEventArgs that contains the event data.

OnPreviewLocationChanged(Point)

Raises the PreviewLocationChanged event and sets the IsPreviewingLocation property to true.

```
protected void OnPreviewLocationChanged(Point newLocation)
```

Parameters

`newLocation` Point

The new location.

OnRenderSizeChanged(SizeChangedInfo)

Raises the SizeChanged event, using the specified information as part of the eventual event data.

```
protected override void OnRenderSizeChanged(SizeChangedInfo sizeInfo)
```

Parameters

`sizeInfo` SizeChangedInfo

Details of the old and new size involved in the change.

OnSelectedChanged(bool)

Raises the SelectedEvent or UnselectedEvent based on `newValue`. Called when the IsSelected value is changed.

```
protected void OnSelectedChanged(bool newValue)
```

Parameters

`newValue` bool

True if selected, false otherwise.

PopAllStates()

Pops all states from the container.

```
public void PopAllStates()
```

Remarks

It doesn't pop the initial state. (see GetInitialState())

PopState()

Pops the current State from the stack.

```
public void PopState()
```

Remarks

It doesn't pop the initial state. (see GetInitialState())
Calls Exit() on the current state.
Calls ReEnter(ContainerState) on the previous state.

PushState(ContainerState)

Pushes the given state to the stack.

```
public void PushState(ContainerState state)
```

Parameters

`state` ContainerState

The new state of the container.

Remarks

Calls Enter(ContainerState?) on the new state.

Events

DragCompleted

Occurs when this ItemContainer completed the drag operation.

```
public event DragCompletedEventHandler DragCompleted
```

Event Type

DragCompletedEventHandler

DragDelta

Occurs when this ItemContainer is being dragged.

```
public event DragDeltaEventHandler DragDelta
```

Event Type

DragDeltaEventHandler

DragStarted

Occurs when this ItemContainer is the instigator of a drag operation.

```
public event DragStartedEventHandler DragStarted
```

Event Type

DragStartedEventHandler

LocationChanged

Occurs when the Location of this ItemContainer is changed.

```
public event RoutedEventHandler LocationChanged
```

Event Type

RoutedEventHandler

PreviewLocationChanged

Occurs when the ItemContainer is previewing a new location.

```
public event PreviewLocationChanged? PreviewLocationChanged
```

Event Type

PreviewLocationChanged

Selected

Occurs when this ItemContainer is selected.

```
public event RoutedEventHandler Selected
```

Event Type

RoutedEventHandler

Unselected

Occurs when this ItemContainer is unselected.

```
public event RoutedEventHandler Unselected
```

Event Type

RoutedEventHandler

Class KnotNode

Namespace: Nodify

Assembly: Nodify.dll

Represents a control that owns a Connector.

```
public class KnotNode : ContentControl
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← KnotNode

Class LineConnection

Namespace: Nodify

Assembly: Nodify.dll

Represents a line that has an arrow indicating its Direction.

```
public class LineConnection : BaseConnection
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Shape ← BaseConnection ← LineConnection

Derived

CircuitConnection

Inherited Members

BaseConnection.SourceProperty , BaseConnection.TargetProperty , BaseConnection.SourceOffsetProperty , BaseConnection.TargetOffsetProperty , BaseConnection.SourceOffsetModeProperty , BaseConnection.TargetOffsetModeProperty , BaseConnection.SourceOrientationProperty , BaseConnection.TargetOrientationProperty , BaseConnection.DirectionProperty , BaseConnection.DirectionalArrowsCountProperty , BaseConnection.DirectionalArrowsOffsetProperty , BaseConnection.SpacingProperty , BaseConnection.ArrowSizeProperty , BaseConnection.ArrowEndsProperty , BaseConnection.ArrowShapeProperty , BaseConnection.SplitCommandProperty , BaseConnection.DisconnectCommandProperty , BaseConnection.ForegroundProperty , BaseConnection.TextProperty , BaseConnection.FontSizeProperty , BaseConnection.FontFamilyProperty , BaseConnection.FontWeightProperty , BaseConnection.FontStyleProperty , BaseConnection.FontStretchProperty , BaseConnection.Source , BaseConnection.Target , BaseConnection.SourceOffset , BaseConnection.TargetOffset , BaseConnection.SourceOffsetMode , BaseConnection.TargetOffsetMode , BaseConnection.SourceOrientation , BaseConnection.TargetOrientation , BaseConnection.Direction , BaseConnection.DirectionalArrowsCount , BaseConnection.DirectionalArrowsOffset , BaseConnection.ArrowEnds , BaseConnection.ArrowShape , BaseConnection.Spacing , BaseConnection.ArrowSize , BaseConnection.SplitCommand , BaseConnection.DisconnectCommand , BaseConnection.Foreground , BaseConnection.Text , BaseConnection.FontSize , BaseConnection.FontFamily , BaseConnection.FontStyle , BaseConnection.FontWeight , BaseConnection.FontStretch , BaseConnection.DisconnectEvent , BaseConnection.SplitEvent , BaseConnection.Disconnect , BaseConnection.Split , BaseConnection.ZeroVector , BaseConnection.DefiningGeometry ,

BaseConnection.DrawDirectionalArrowheadGeometry(StreamGeometryContext, Vector, Point) , BaseConnection.DrawArrowGeometry(StreamGeometryContext, Point, Point, ConnectionDirection, ArrowHeadShape, Orientation) , BaseConnection.DrawRectangleArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.DrawEllipseArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation) , BaseConnection.GetOffset() , BaseConnection.GetTextPosition(FormattedText, Point, Point) , BaseConnection.StartAnimation(double) , BaseConnection.StopAnimation() , BaseConnection.OnMouseDown(MouseButtonEventArgs) , BaseConnection.OnMouseUp(MouseButtonEventArgs) , BaseConnection.OnRender(DrawingContext)

Methods

DrawDefaultArrowhead(StreamGeometryContext, Point, Point, ConnectionDirection, Orientation)

```
protected override void DrawDefaultArrowhead(StreamGeometryContext context, Point source,
Point target, ConnectionDirection arrowDirection = ConnectionDirection.Forward, Orientation
orientation = Orientation.Horizontal)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

`arrowDirection` ConnectionDirection

`orientation` Orientation

DrawDirectionalArrowsGeometry(StreamGeometryContext, Point, Point)

```
protected override void DrawDirectionalArrowsGeometry(StreamGeometryContext context, Point
source, Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

DrawLineGeometry(StreamGeometryContext, Point, Point)

```
protected override ((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource,
Point ArrowEndTarget)) DrawLineGeometry(StreamGeometryContext context, Point source,
Point target)
```

Parameters

`context` StreamGeometryContext

`source` Point

`target` Point

Returns

((Point ArrowStartSource, Point ArrowStartTarget), (Point ArrowEndSource, Point ArrowEndTarget))

InterpolateLineSegment(Point, Point, double)

```
protected static Point InterpolateLineSegment(Point p0, Point p1, double t)
```

Parameters

`p0` Point

`p1` Point

`t` double

Returns

Point

Class MultiGesture

Namespace: Nodify

Assembly: Nodify.dll

Combines multiple input gestures.

```
public class MultiGesture : InputGesture
```

Inheritance

object ← InputGesture ← MultiGesture

Derived

AllGestures, AnyGesture

Constructors

MultiGesture(Match, params InputGesture[])

Constructs an instance of a MultiGesture.

```
public MultiGesture(MultiGesture.Match match, params InputGesture[] gestures)
```

Parameters

`match` MultiGesture.Match

The matching strategy.

`gestures` InputGesture[]

The input gestures.

Fields

None

```
public static readonly MultiGesture None
```

Field Value

MultiGesture

Methods

Matches(object, InputEventArgs)

When overridden in a derived class, determines whether the specified InputGesture matches the input associated with the specified InputEventArgs object.

```
public override bool Matches(object targetElement, InputEventArgs inputEventArgs)
```

Parameters

`targetElement` object

The target of the command.

`inputEventArgs` InputEventArgs

The input event data to compare this gesture to.

Returns

bool

true if the gesture matches the input; otherwise, false.

Enum MultiGesture.Match

Namespace: Nodify

Assembly: Nodify.dll

The strategy used by Matches(object, InputEventArgs).

```
public enum MultiGesture.Match
```

Fields

`All = 1`

All gestures must match.

`Any = 0`

At least one gesture must match.

Class Node

Namespace: Nodify

Assembly: Nodify.dll

Represents a control that has a list of InputConnectors and a list of OutputConnectors.

```
public class Node : HeaderedContentControl
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← HeaderedContentControl ← Node

Fields

ContentBrushProperty

```
public static readonly DependencyProperty ContentBrushProperty
```

Field Value

DependencyProperty

FooterBrushProperty

```
public static readonly DependencyProperty FooterBrushProperty
```

Field Value

DependencyProperty

FooterProperty

```
public static readonly DependencyProperty FooterProperty
```

Field Value

DependencyProperty

FooterTemplateProperty

```
public static readonly DependencyProperty FooterTemplateProperty
```

Field Value

DependencyProperty

HasFooterProperty

```
public static readonly DependencyProperty HasFooterProperty
```

Field Value

DependencyProperty

HasFooterPropertyKey

```
protected static readonly DependencyPropertyKey HasFooterPropertyKey
```

Field Value

DependencyPropertyKey

HeaderBrushProperty

```
public static readonly DependencyProperty HeaderBrushProperty
```

Field Value

DependencyProperty

InputConnectorTemplateProperty

```
public static readonly DependencyProperty InputConnectorTemplateProperty
```

Field Value

DependencyProperty

InputProperty

```
public static readonly DependencyProperty InputProperty
```

Field Value

DependencyProperty

OutputConnectorTemplateProperty

```
public static readonly DependencyProperty OutputConnectorTemplateProperty
```

Field Value

DependencyProperty

OutputProperty

```
public static readonly DependencyProperty OutputProperty
```

Field Value

DependencyProperty

Properties

ContentBrush

Gets or sets the brush used for the background of the Content of this Node.

```
public Brush ContentBrush { get; set; }
```

Property Value

Brush

Footer

Gets or sets the data for the footer of this control.

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

Property Value

object

FooterBrush

Gets or sets the brush used for the background of the Footer of this Node.

```
public Brush FooterBrush { get; set; }
```

Property Value

Brush

FooterTemplate

Gets or sets the template used to display the content of the control's footer.

```
public DataTemplate FooterTemplate { get; set; }
```

Property Value

DataTemplate

HasFooter

Gets a value that indicates whether the Footer is null.

```
public bool HasFooter { get; }
```

Property Value

bool

HeaderBrush

Gets or sets the brush used for the background of the Header of this Node.

```
public Brush HeaderBrush { get; set; }
```

Property Value

Brush

Input

Gets or sets the data for the input Connectors of this control.

```
public IEnumerable Input { get; set; }
```

Property Value

IEnumerable

InputConnectorTemplate

Gets or sets the template used to display the content of the control's Input connectors.

```
public DataTemplate InputConnectorTemplate { get; set; }
```

Property Value

DataTemplate

Output

Gets or sets the data for the output Connectors of this control.

```
public IEnumerable Output { get; set; }
```

Property Value

IEnumerable

OutputConnectorTemplate

Gets or sets the template used to display the content of the control's Output connectors.

```
public DataTemplate OutputConnectorTemplate { get; set; }
```

Property Value

DataTemplate

Class NodeInput

Namespace: Nodify

Assembly: Nodify.dll

Represents the default control for the InputConnectorTemplate.

```
public class NodeInput : Connector
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← Connector ← NodeInput

Inherited Members

Connector.ElementConnector , Connector.PendingConnectionStartedEvent , Connector.PendingConnectionCompletedEvent , Connector.PendingConnectionDragEvent , Connector.DisconnectEvent , Connector.PendingConnectionStarted , Connector.PendingConnectionCompleted , Connector.PendingConnectionDrag , Connector.Disconnect , Connector.AnchorProperty , Connector.IsConnectedProperty , Connector.DisconnectCommandProperty , Connector.IsPendingConnectionProperty , Connector.Anchor , Connector.IsConnected , Connector.IsPendingConnection , Connector.DisconnectCommand , Connector.Thumb , Connector.Container , Connector.Editor , Connector.OptimizeSafeZone , Connector.OptimizeMinimumSelectedItems , Connector.EnableOptimizations , Connector.AllowPendingConnectionCancellation , Connector.EnableStickyConnections , Connector.OnApplyTemplate() , Connector.OnRenderSizeChanged(SizeChangedInfo) , Connector.UpdateAnchorOptimized(Point) , Connector.UpdateAnchor(Point) , Connector.UpdateAnchor() , Connector.OnLostMouseCapture(MouseEventArgs) , Connector.OnMouseDown(MouseButtonEventArgs) , Connector.OnMouseUp(MouseButtonEventArgs) , Connector.OnKeyUp(KeyEventArgs) , Connector.OnMouseMove(MouseEventArgs) , Connector.OnConnectorDrag(Vector) , Connector.OnConnectorDragStarted() , Connector.OnConnectorDragCompleted(bool) , Connector.OnDisconnect()

Fields

ConnectorTemplateProperty

```
public static readonly DependencyProperty ConnectorTemplateProperty
```

Field Value

DependencyProperty

HeaderProperty

```
public static readonly DependencyProperty HeaderProperty
```

Field Value

DependencyProperty

HeaderTemplateProperty

```
public static readonly DependencyProperty HeaderTemplateProperty
```

Field Value

DependencyProperty

OrientationProperty

```
public static readonly DependencyProperty OrientationProperty
```

Field Value

DependencyProperty

Properties

ConnectorTemplate

Gets or sets the template used to display the connecting point of this Connector.

```
public ControlTemplate ConnectorTemplate { get; set; }
```

Property Value

ControlTemplate

Header

Gets of sets the data used for the control's header.

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

Property Value

object

HeaderTemplate

Gets or sets the template used to display the content of the control's header.

```
public DataTemplate HeaderTemplate { get; set; }
```

Property Value

DataTemplate

Orientation

Gets or sets a value that indicates the dimension by which child elements are stacked.

```
public Orientation Orientation { get; set; }
```

Property Value

Orientation

The Orientation of child content.

Class NodeOutput

Namespace: Nodify

Assembly: Nodify.dll

Represents the default control for the OutputConnectorTemplate.

```
public class NodeOutput : Connector
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← Connector ← NodeOutput

Inherited Members

Connector.ElementConnector , Connector.PendingConnectionStartedEvent , Connector.PendingConnectionCompletedEvent , Connector.PendingConnectionDragEvent , Connector.DisconnectEvent , Connector.PendingConnectionStarted , Connector.PendingConnectionCompleted , Connector.PendingConnectionDrag , Connector.Disconnect , Connector.AnchorProperty , Connector.IsConnectedProperty , Connector.DisconnectCommandProperty , Connector.IsPendingConnectionProperty , Connector.Anchor , Connector.IsConnected , Connector.IsPendingConnection , Connector.DisconnectCommand , Connector.Thumb , Connector.Container , Connector.Editor , Connector.OptimizeSafeZone , Connector.OptimizeMinimumSelectedItems , Connector.EnableOptimizations , Connector.AllowPendingConnectionCancellation , Connector.EnableStickyConnections , Connector.OnApplyTemplate() , Connector.OnRenderSizeChanged(SizeChangedInfo) , Connector.UpdateAnchorOptimized(Point) , Connector.UpdateAnchor(Point) , Connector.UpdateAnchor() , Connector.OnLostMouseCapture(MouseEventArgs) , Connector.OnMouseDown(MouseButtonEventArgs) , Connector.OnMouseUp(MouseButtonEventArgs) , Connector.OnKeyUp(KeyEventArgs) , Connector.OnMouseMove(MouseEventArgs) , Connector.OnConnectorDrag(Vector) , Connector.OnConnectorDragStarted() , Connector.OnConnectorDragCompleted(bool) , Connector.OnDisconnect()

Fields

ConnectorTemplateProperty

```
public static readonly DependencyProperty ConnectorTemplateProperty
```

Field Value

DependencyProperty

HeaderProperty

```
public static readonly DependencyProperty HeaderProperty
```

Field Value

DependencyProperty

HeaderTemplateProperty

```
public static readonly DependencyProperty HeaderTemplateProperty
```

Field Value

DependencyProperty

OrientationProperty

```
public static readonly DependencyProperty OrientationProperty
```

Field Value

DependencyProperty

Properties

ConnectorTemplate

Gets or sets the template used to display the connecting point of this Connector.

```
public ControlTemplate ConnectorTemplate { get; set; }
```

Property Value

ControlTemplate

Header

Gets of sets the data used for the control's header.

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

Property Value

object

HeaderTemplate

Gets or sets the template used to display the content of the control's header.

```
public DataTemplate HeaderTemplate { get; set; }
```

Property Value

DataTemplate

Orientation

Gets or sets a value that indicates the dimension by which child elements are stacked.

```
public Orientation Orientation { get; set; }
```

Property Value

Orientation

The Orientation of child content.

Class NodifyCanvas

Namespace: Nodify

Assembly: Nodify.dll

A canvas like panel that works with INodifyCanvasItems.

```
public class NodifyCanvas : Panel
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Panel ← NodifyCanvas

Fields

ExtentProperty

```
public static readonly DependencyProperty ExtentProperty
```

Field Value

DependencyProperty

Properties

Extent

The area covered by the children of this panel.

```
public Rect Extent { get; set; }
```

Property Value

Rect

Methods

ArrangeOverride(Size)

When overridden in a derived class, positions child elements and determines a size for a Framework Element derived class.

```
protected override Size ArrangeOverride(Size arrangeSize)
```

Parameters

`arrangeSize` Size

Returns

Size

The actual size used.

MeasureOverride(Size)

When overridden in a derived class, measures the size in layout required for child elements and determines a size for the FrameworkElement-derived class.

```
protected override Size MeasureOverride(Size constraint)
```

Parameters

`constraint` Size

Returns

Size

The size that this element determines it needs during layout, based on its calculations of child element sizes.

Class NodifyEditor

Namespace: Nodify

Assembly: Nodify.dll

Groups ItemContainers and Connections in an area that you can drag, zoom and select.

```
[TemplatePart(Name = "PART_ItemsHost", Type = typeof(Panel))]
[StyleTypedProperty(Property = "ItemContainerStyle", StyleTargetType
= typeof(ItemContainer))]
[StyleTypedProperty(Property = "DecoratorContainerStyle", StyleTargetType
= typeof(DecoratorContainer))]
[StyleTypedProperty(Property = "SelectionRectangleStyle", StyleTargetType
= typeof(Rectangle))]
public class NodifyEditor : MultiSelector
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ItemsControl ← Selector ← MultiSelector ← NodifyEditor

Constructors

NodifyEditor()

Initializes a new instance of the NodifyEditor class.

```
public NodifyEditor()
```

Fields

AutoPanEdgeDistanceProperty

```
public static readonly DependencyProperty AutoPanEdgeDistanceProperty
```

Field Value

DependencyProperty

AutoPanSpeedProperty

```
public static readonly DependencyProperty AutoPanSpeedProperty
```

Field Value

DependencyProperty

BringIntoViewMaxDurationProperty

```
public static readonly DependencyProperty BringIntoViewMaxDurationProperty
```

Field Value

DependencyProperty

BringIntoViewSpeedProperty

```
public static readonly DependencyProperty BringIntoViewSpeedProperty
```

Field Value

DependencyProperty

ConnectionCompletedCommandProperty

```
public static readonly DependencyProperty ConnectionCompletedCommandProperty
```

Field Value

DependencyProperty

ConnectionStartedCommandProperty

```
public static readonly DependencyProperty ConnectionStartedCommandProperty
```

Field Value

DependencyProperty

ConnectionTemplateProperty

```
public static readonly DependencyProperty ConnectionTemplateProperty
```

Field Value

DependencyProperty

ConnectionsProperty

```
public static readonly DependencyProperty ConnectionsProperty
```

Field Value

DependencyProperty

DecoratorContainerStyleProperty

```
public static readonly DependencyProperty DecoratorContainerStyleProperty
```

Field Value

DependencyProperty

DecoratorTemplateProperty

```
public static readonly DependencyProperty DecoratorTemplateProperty
```

Field Value

DependencyProperty

DecoratorsExtentProperty

```
public static readonly DependencyProperty DecoratorsExtentProperty
```

Field Value

DependencyProperty

DecoratorsProperty

```
public static readonly DependencyProperty DecoratorsProperty
```

Field Value

DependencyProperty

DisableAutoPanningProperty

```
public static readonly DependencyProperty DisableAutoPanningProperty
```

Field Value

DependencyProperty

DisablePanningProperty

```
public static readonly DependencyProperty DisablePanningProperty
```

Field Value

DependencyProperty

DisableZoomingProperty

```
public static readonly DependencyProperty DisableZoomingProperty
```

Field Value

DependencyProperty

DisconnectConnectorCommandProperty

```
public static readonly DependencyProperty DisconnectConnectorCommandProperty
```

Field Value

DependencyProperty

DisplayConnectionsOnTopProperty

```
public static readonly DependencyProperty DisplayConnectionsOnTopProperty
```

Field Value

DependencyProperty

ElementItemsHost

```
protected const string ElementItemsHost = "PART_ItemsHost"
```

Field Value

string

EnableRealtimeSelectionProperty

```
public static readonly DependencyProperty EnableRealtimeSelectionProperty
```

Field Value

DependencyProperty

GridCellSizeProperty

```
public static readonly DependencyProperty GridCellSizeProperty
```

Field Value

DependencyProperty

IsPanningProperty

```
public static readonly DependencyProperty IsPanningProperty
```

Field Value

DependencyProperty

IsPanningPropertyKey

```
public static readonly DependencyPropertyKey IsPanningPropertyKey
```

Field Value

DependencyPropertyKey

IsSelectingProperty

```
public static readonly DependencyProperty IsSelectingProperty
```

Field Value

DependencyProperty

IsSelectingPropertyKey

```
protected static readonly DependencyPropertyKey IsSelectingPropertyKey
```

Field Value

DependencyPropertyKey

ItemsDragCompletedCommandProperty

```
public static readonly DependencyProperty ItemsDragCompletedCommandProperty
```

Field Value

DependencyProperty

ItemsDragStartedCommandProperty

```
public static readonly DependencyProperty ItemsDragStartedCommandProperty
```

Field Value

DependencyProperty

ItemsExtentProperty

```
public static readonly DependencyProperty ItemsExtentProperty
```

Field Value

DependencyProperty

ItemsSelectCompletedCommandProperty

```
public static readonly DependencyProperty ItemsSelectCompletedCommandProperty
```

Field Value

DependencyProperty

ItemsSelectStartedCommandProperty

```
public static readonly DependencyProperty ItemsSelectStartedCommandProperty
```

Field Value

DependencyProperty

MaxViewportZoomProperty

```
public static readonly DependencyProperty MaxViewportZoomProperty
```

Field Value

DependencyProperty

MinViewportZoomProperty

```
public static readonly DependencyProperty MinViewportZoomProperty
```

Field Value

DependencyProperty

MouseLocationProperty

```
public static readonly DependencyProperty MouseLocationProperty
```

Field Value

DependencyProperty

MouseLocationPropertyKey

```
protected static readonly DependencyPropertyKey MouseLocationPropertyKey
```

Field Value

DependencyPropertyKey

PendingConnectionProperty

```
public static readonly DependencyProperty PendingConnectionProperty
```

Field Value

DependencyProperty

PendingConnectionTemplateProperty

```
public static readonly DependencyProperty PendingConnectionTemplateProperty
```

Field Value

DependencyProperty

RemoveConnectionCommandProperty

```
public static readonly DependencyProperty RemoveConnectionCommandProperty
```

Field Value

DependencyProperty

ScaleTransform

Gets the transform used to zoom on the viewport.

```
protected readonly ScaleTransform ScaleTransform
```

Field Value

ScaleTransform

SelectedAreaProperty

```
public static readonly DependencyProperty SelectedAreaProperty
```

Field Value

DependencyProperty

SelectedAreaPropertyKey

```
protected static readonly DependencyPropertyKey SelectedAreaPropertyKey
```

Field Value

DependencyPropertyKey

SelectedItemsProperty

```
public static readonly DependencyProperty SelectedItemsProperty
```

Field Value

DependencyProperty

SelectionRectangleStyleProperty

```
public static readonly DependencyProperty SelectionRectangleStyleProperty
```

Field Value

DependencyProperty

TranslateTransform

Gets the transform used to offset the viewport.

```
protected readonly TranslateTransform TranslateTransform
```

Field Value

TranslateTransform

ViewportLocationProperty

```
public static readonly DependencyProperty ViewportLocationProperty
```

Field Value

DependencyProperty

ViewportSizeProperty

```
public static readonly DependencyProperty ViewportSizeProperty
```

Field Value

DependencyProperty

ViewportTransformProperty

```
public static readonly DependencyProperty ViewportTransformProperty
```

Field Value

DependencyProperty

ViewportTransformPropertyKey

```
protected static readonly DependencyPropertyKey ViewportTransformPropertyKey
```

Field Value

DependencyPropertyKey

ViewportUpdatedEvent

```
public static readonly RoutedEvent ViewportUpdatedEvent
```

Field Value

RoutedEvent

ViewportZoomProperty

```
public static readonly DependencyProperty ViewportZoomProperty
```

Field Value

DependencyProperty

Properties

AutoPanEdgeDistance

Gets or sets the maximum distance in pixels from the edge of the editor that will trigger auto-panning.

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

Property Value

double

AutoPanSpeed

Gets or sets the speed used when auto-panning scaled by AutoPanningTickRate

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

Property Value

double

AutoPanningTickRate

Gets or sets how often the new ViewportLocation is calculated in milliseconds when DisableAutoPanning is false.

```
public static double AutoPanningTickRate { get; set; }
```

Property Value

double

BringIntoViewMaxDuration

Gets or sets the maximum animation duration in seconds for bringing a location into view.

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

Property Value

double

BringIntoViewSpeed

Gets or sets the animation speed in pixels per second for bringing a location into view.

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

Property Value

double

Remarks

Total animation duration is calculated based on distance and clamped between 0.1 and BringIntoView MaxDuration.

ConnectionCompletedCommand

Invoked when the PendingConnection is completed.
Use CompletedCommand if you want to control the visibility of the connection from the viewmodel.
Parameter is Tuple<T1, T2> where Item1 is the Source and Item2 is Target.

```
public ICommand? ConnectionCompletedCommand { get; set; }
```

Property Value

ICommand

ConnectionStartedCommand

Invoked when the PendingConnection is completed.
Use StartedCommand if you want to control the visibility of the connection from the viewmodel.
Parameter is Source.

```
public ICommand? ConnectionStartedCommand { get; set; }
```

Property Value

ICommand

ConnectionTemplate

Gets or sets the DataTemplate to use when generating a new BaseConnection.

```
public DataTemplate ConnectionTemplate { get; set; }
```

Property Value

DataTemplate

Connections

Gets or sets the data source that BaseConnections will be generated for.

```
public IEnumerable Connections { get; set; }
```

Property Value

IEnumerable

DecoratorContainerStyle

Gets or sets the style to use for the DecoratorContainer.

```
public Style DecoratorContainerStyle { get; set; }
```

Property Value

Style

DecoratorTemplate

Gets or sets the DataTemplate to use when generating a new DecoratorContainer.

```
public DataTemplate DecoratorTemplate { get; set; }
```

Property Value

DataTemplate

Decorators

Gets or sets the items that will be rendered in the decorators layer via DecoratorContainers.

```
public IEnumerable Decorators { get; set; }
```

Property Value

IEnumerable

DecoratorsExtent

The area covered by the DecoratorContainers.

```
public Rect DecoratorsExtent { get; set; }
```

Property Value

Rect

DisableAutoPanning

Gets or sets whether to disable the auto panning when selecting or dragging near the edge of the editor configured by AutoPanEdgeDistance.

```
public bool DisableAutoPanning { get; set; }
```

Property Value

bool

DisablePanning

Gets or sets whether panning should be disabled.

```
public bool DisablePanning { get; set; }
```

Property Value

bool

DisableZooming

Gets or sets whether zooming should be disabled.

```
public bool DisableZooming { get; set; }
```

Property Value

bool

DisconnectConnectorCommand

Invoked when the Disconnect event is raised.
Can also be handled at the Connector level using the DisconnectCommand command.
Parameter is the Connector's DataContext.

```
public ICommand? DisconnectConnectorCommand { get; set; }
```

Property Value

ICommand

DisplayConnectionsOnTop

Gets or sets whether to display connections on top of ItemContainers or not.

```
public bool DisplayConnectionsOnTop { get; set; }
```

Property Value

bool

EnableDraggingContainersOptimizations

Gets or sets if the current position of containers that are being dragged should not be committed until the end of the dragging operation.

```
public static bool EnableDraggingContainersOptimizations { get; set; }
```

Property Value

bool

EnableRealtimeSelection

Enables selecting and deselecting items while the SelectedArea changes. Disable for maximum performance when hundreds of items are generated.

```
public bool EnableRealtimeSelection { get; set; }
```

Property Value

bool

EnableRenderingContainersOptimizations

Gets or sets if NodifyEditors should enable optimizations based on OptimizeRenderingMinimumContainers and OptimizeRenderingZoomOutPercent.

```
public static bool EnableRenderingContainersOptimizations { get; set; }
```

Property Value

bool

EnableSnappingCorrection

Correct ItemContainer's position after moving if starting position is not snapped to grid.

```
public static bool EnableSnappingCorrection { get; set; }
```

Property Value

bool

FitToScreenExtentMargin

Gets or sets the margin to add in all directions to the ItemsExtent or area parameter when using FitToScreen(Rect?).

```
public static double FitToScreenExtentMargin { get; set; }
```

Property Value

double

GridCellSize

Gets or sets the value of an invisible grid used to adjust locations (snapping) of ItemContainers.

```
public uint GridCellSize { get; set; }
```

Property Value

uint

HandleRightClickAfterPanningThreshold

Gets or sets the maximum number of pixels allowed to move the mouse before cancelling the mouse event. Useful for ContextMenus to appear if mouse only moved a bit or not at all.

```
public static double HandleRightClickAfterPanningThreshold { get; set; }
```

Property Value

double

IsBulkUpdatingItems

Tells if the NodifyEditor is doing operations on multiple items at once.

```
public bool IsBulkUpdatingItems { get; protected set; }
```

Property Value

bool

IsPanning

Gets a value that indicates whether a panning operation is in progress.

```
public bool IsPanning { get; protected set; }
```

Property Value

bool

IsSelecting

Gets a value that indicates whether a selection operation is in progress.

```
public bool IsSelecting { get; }
```

Property Value

bool

ItemsDragCompletedCommand

Invoked when a drag operation is completed for the SelectedItems.

```
public ICommand? ItemsDragCompletedCommand { get; set; }
```

Property Value

ICommand

ItemsDragStartedCommand

Invoked when a drag operation starts for the SelectedItems.

```
public ICommand? ItemsDragStartedCommand { get; set; }
```

Property Value

ICommand

ItemsExtent

The area covered by the ItemContainers.

```
public Rect ItemsExtent { get; set; }
```

Property Value

Rect

ItemsHost

Gets the panel that holds all the ItemContainers.

```
protected Panel ItemsHost { get; }
```

Property Value

Panel

ItemsSelectCompletedCommand

Invoked when a selection operation is completed.

```
public ICommand? ItemsSelectCompletedCommand { get; set; }
```

Property Value

ICommand

ItemsSelectStartedCommand

Invoked when a selection operation is started.

```
public ICommand? ItemsSelectStartedCommand { get; set; }
```

Property Value

ICommand

MaxViewportZoom

Gets or sets the maximum zoom factor of the viewport

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

Property Value

double

MinViewportZoom

Gets or sets the minimum zoom factor of the viewport

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

Property Value

double

MouseLocation

Gets the current mouse location in graph space coordinates (relative to the ItemsHost).

```
public Point MouseLocation { get; protected set; }
```

Property Value

Point

OptimizeRenderingMinimumContainers

Gets or sets the minimum number of ItemContainers needed to trigger optimizations when reaching the OptimizeRenderingZoomOutPercent.

```
public static uint OptimizeRenderingMinimumContainers { get; set; }
```

Property Value

uint

OptimizeRenderingZoomOutPercent

Gets or sets the minimum zoom out percent needed to start optimizing the rendering for ItemContainers. Value is between 0 and 1.

```
public static double OptimizeRenderingZoomOutPercent { get; set; }
```

Property Value

double

PendingConnection

Gets of sets the DataContext of the PendingConnection.

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

Property Value

object

PendingConnectionTemplate

Gets or sets the DataTemplate to use for the PendingConnection.

```
public DataTemplate PendingConnectionTemplate { get; set; }
```

Property Value

DataTemplate

RemoveConnectionCommand

Invoked when the Disconnect event is raised.
Can also be handled at the BaseConnection level using the DisconnectCommand command.
Parameter is the BaseConnection's DataContext.

```
public ICommand? RemoveConnectionCommand { get; set; }
```

Property Value

ICommand

SelectedArea

Gets the currently selected area while IsSelecting is true.

```
public Rect SelectedArea { get; }
```

Property Value

Rect

SelectedContainers

Gets a list of ItemContainers that are selected.

```
protected IReadOnlyList<ItemContainer> SelectedContainers { get; }
```

Property Value

IReadOnlyList<ItemContainer>

Remarks

Cache the result before using it to avoid extra allocations.

SelectedItems

Gets or sets the items in the NodifyEditor that are selected.

```
public IList? SelectedItems { get; set; }
```

Property Value

IList

SelectionRectangleStyle

Gets or sets the style to use for the selection rectangle.

```
public Style SelectionRectangleStyle { get; set; }
```

Property Value

Style

State

The current state of the editor.

```
public EditorState State { get; }
```

Property Value

EditorState

ViewportLocation

Gets or sets the viewport's top-left coordinates in graph space coordinates.

```
public Point ViewportLocation { get; set; }
```

Property Value

Point

ViewportSize

Gets the size of the viewport.

```
public Size ViewportSize { get; set; }
```

Property Value

Size

ViewportTransform

Gets the transform that is applied to all child controls.

```
public Transform ViewportTransform { get; }
```

Property Value

Transform

ViewportZoom

Gets or sets the zoom factor of the viewport.

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

Property Value

double

Methods

BringIntoView(Point, bool, Action?)

Moves the viewport center at the specified location.

```
public void BringIntoView(Point point, bool animated = true, Action? onFinish = null)
```

Parameters

`point` Point

The location in graph space coordinates.

`animated` bool

True to animate the movement.

`onFinish` Action

The callback invoked when movement is finished.

Remarks

Temporarily disables editor controls when animated.

FitToScreen(Rect?)

Scales the viewport to fit the specified `area` or all the ItemContainers if that's possible.

```
public void FitToScreen(Rect? area = null)
```

Parameters

`area` Rect?

Remarks

Does nothing if `area` is null and there's no items.

GetContainerForItemOverride()

Creates or identifies the element that is used to display the given item.

```
protected override DependencyObject GetContainerForItemOverride()
```

Returns

DependencyObject

The element that is used to display the given item.

GetInitialState()

Creates the initial state of the editor.

```
protected virtual EditorState GetInitialState()
```

Returns

EditorState

The initial state.

GetLocationInsideEditor(DragEventArgs)

Translates the event location to graph space coordinates (relative to the ItemsHost).

```
public Point GetLocationInsideEditor(DragEventArgs args)
```

Parameters

`args` DragEventArgs

The drag event.

Returns

Point

A location inside the graph

GetLocationInsideEditor(MouseEventArgs)

Translates the event location to graph space coordinates (relative to the ItemsHost).

```
public Point GetLocationInsideEditor(MouseEventArgs args)
```

Parameters

`args` MouseEventArgs

The mouse event.

Returns

Point

A location inside the graph

GetLocationInsideEditor(Point, UIElement)

Translates the specified location to graph space coordinates (relative to the ItemsHost).

```
public Point GetLocationInsideEditor(Point location, UIElement relativeTo)
```

Parameters

`location` Point

The location coordinates relative to `relativeTo`

`relativeTo` UIElement

The element where the `location` was calculated from.

Returns

Point

A location inside the graph.

InvertSelection(Rect, bool)

Inverts the ItemContainers selection in the specified `area`.

```
public void InvertSelection(Rect area, bool fit = false)
```

Parameters

`area` Rect

The area to look for ItemContainers.

`fit` bool

True to check if the `area` contains the ItemContainer.
False to check if `area` intersects the ItemContainer.

IsItemItsOwnContainerOverride(object)

Determines if the specified item is (or is eligible to be) its own container.

```
protected override bool IsItemItsOwnContainerOverride(object item)
```

Parameters

`item` object

The item to check.

Returns

bool

true if the item is (or is eligible to be) its own container; otherwise, false.

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

OnDisableAutoPanningChanged(bool)

Called when the DisableAutoPanning changes.

```
protected virtual void OnDisableAutoPanningChanged(bool shouldDisable)
```

Parameters

`shouldDisable` bool

Whether to enable or disable auto panning.

OnKeyDown(KeyEventArgs)

Invoked when the KeyDown event is received.

```
protected override void OnKeyDown(KeyEventArgs e)
```

Parameters

e KeyEventArgs

Information about the event.

OnKeyUp(KeyEventArgs)

Invoked when an unhandled System.Windows.Input.Keyboard.KeyUp attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnKeyUp(KeyEventArgs e)
```

Parameters

e KeyEventArgs

The KeyEventArgs that contains the event data.

OnLostMouseCapture(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.LostMouseCapture attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnLostMouseCapture(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains event data.

OnMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

OnMouseMove(MouseEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseMove attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseMove(MouseEventArgs e)
```

Parameters

e MouseEventArgs

The MouseEventArgs that contains the event data.

OnMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.

OnMouseWheel(MouseWheelEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseWheel attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseWheel(MouseWheelEventArgs e)
```

Parameters

e MouseWheelEventArgs

The MouseWheelEventArgs that contains the event data.

OnRenderSizeChanged(SizeChangedInfo)

Raises the SizeChanged event, using the specified information as part of the eventual event data.

```
protected override void OnRenderSizeChanged(SizeChangedInfo sizeInfo)
```

Parameters

sizeInfo SizeChangedInfo

Details of the old and new size involved in the change.

OnSelectionChanged(SelectionChangedEventArgs)

Called when the selection changes.

```
protected override void OnSelectionChanged(SelectionChangedEventArgs e)
```

Parameters

e SelectionChangedEventArgs

The event data.

OnViewportUpdated()

Updates the ViewportSize and raises the ViewportUpdatedEvent. Called when the RenderSize or ViewportZoom is changed.

```
protected void OnViewportUpdated()
```

PopAllStates()

Pops all states from the editor.

```
public void PopAllStates()
```

Remarks

It doesn't pop the initial state. (see GetInitialState())

PopState()

Pops the current State from the stack.

```
public void PopState()
```

Remarks

It doesn't pop the initial state. (see GetInitialState())
Calls Exit() on the current state.
Calls ReEnter(EditorState) on the previous state.

PushState(EditorState)

Pushes the given state to the stack.

```
public void PushState(EditorState state)
```

Parameters

`state` EditorState

The new state of the editor.

Remarks

Calls Enter(EditorState?) on the new state.

SelectArea(Rect, bool, bool)

Selects the ItemContainers in the specified `area`.

```
public void SelectArea(Rect area, bool append = false, bool fit = false)
```

Parameters

`area` Rect

The area to look for ItemContainers.

`append` bool

If true, it will add to the existing selection.

`fit` bool

True to check if the `area` contains the ItemContainer.
False to check if `area` intersects the ItemContainer.

UnselectArea(Rect, bool)

Unselect the ItemContainers in the specified `area`.

```
public void UnselectArea(Rect area, bool fit = false)
```

Parameters

`area` Rect

The area to look for ItemContainers.

`fit` bool

True to check if the `area` contains the ItemContainer.
False to check if `area` intersects the ItemContainer.

ZoomAtPosition(double, Point)

Zoom at the specified location in graph space coordinates.

```
public void ZoomAtPosition(double zoom, Point location)
```

Parameters

`zoom` double

The zoom factor.

`location` Point

The location to focus when zooming.

ZoomIn()

Zoom in at the viewports center

```
public void ZoomIn()
```

ZoomOut()

Zoom out at the viewports center

```
public void ZoomOut()
```

Events

ViewportUpdated

Occurs whenever the viewport updates.

```
public event RoutedEventHandler ViewportUpdated
```

Event Type

RoutedEventHandler

Class PendingConnection

Namespace: Nodify

Assembly: Nodify.dll

Represents a pending connection usually started by a Connector which invokes the CompletedCommand when completed.

```
public class PendingConnection : ContentControl
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← ContentControl ← PendingConnection

Fields

AllowOnlyConnectorsProperty

```
public static readonly DependencyProperty AllowOnlyConnectorsProperty
```

Field Value

DependencyProperty

CompletedCommandProperty

```
public static readonly DependencyProperty CompletedCommandProperty
```

Field Value

DependencyProperty

DirectionProperty

```
public static readonly DependencyProperty DirectionProperty
```

Field Value

DependencyProperty

EnablePreviewProperty

```
public static readonly DependencyProperty EnablePreviewProperty
```

Field Value

DependencyProperty

EnableSnappingProperty

```
public static readonly DependencyProperty EnableSnappingProperty
```

Field Value

DependencyProperty

IsOverElementProperty

Will be set for Connectors and ItemContainers when the pending connection is over the element if EnablePreview or EnableSnapping is true.

```
public static readonly DependencyProperty IsOverElementProperty
```

Field Value

DependencyProperty

IsVisibleProperty

```
public static readonly DependencyProperty IsVisibleProperty
```

Field Value

DependencyProperty

PreviewTargetProperty

```
public static readonly DependencyProperty PreviewTargetProperty
```

Field Value

DependencyProperty

SourceAnchorProperty

```
public static readonly DependencyProperty SourceAnchorProperty
```

Field Value

DependencyProperty

SourceProperty

```
public static readonly DependencyProperty SourceProperty
```

Field Value

DependencyProperty

StartedCommandProperty

```
public static readonly DependencyProperty StartedCommandProperty
```

Field Value

DependencyProperty

StrokeDashArrayProperty

```
public static readonly DependencyProperty StrokeDashArrayProperty
```

Field Value

DependencyProperty

StrokeProperty

```
public static readonly DependencyProperty StrokeProperty
```

Field Value

DependencyProperty

StrokeThicknessProperty

```
public static readonly DependencyProperty StrokeThicknessProperty
```

Field Value

DependencyProperty

TargetAnchorProperty

```
public static readonly DependencyProperty TargetAnchorProperty
```

Field Value

DependencyProperty

TargetProperty

```
public static readonly DependencyProperty TargetProperty
```

Field Value

DependencyProperty

Properties

AllowOnlyConnectors

If true will preview and connect only to Connectors, otherwise will also enable ItemContainers.

```
public bool AllowOnlyConnectors { get; set; }
```

Property Value

bool

CompletedCommand

Gets or sets the command to invoke when the pending connection is completed. Will not be invoked if ConnectionCompletedCommand is used. Target will be set to the desired Connector's DataContext and will also be the command's parameter.

```
public ICommand? CompletedCommand { get; set; }
```

Property Value

ICommand

Direction

Gets or sets the direction of this connection.

```
public ConnectionDirection Direction { get; set; }
```

Property Value

ConnectionDirection

Editor

Gets the NodifyEditor that owns this PendingConnection.

```
protected NodifyEditor? Editor { get; }
```

Property Value

NodifyEditor

EnablePreview

PreviewTarget will be updated with a potential Connector's DataContext if this is true.

```
public bool EnablePreview { get; set; }
```

Property Value

bool

EnableSnapping

Enables snapping the TargetAnchor to a possible Target connector.

```
public bool EnableSnapping { get; set; }
```

Property Value

bool

IsVisible

Gets or sets the visibility of the connection.

```
public bool IsVisible { get; set; }
```

Property Value

bool

PreviewTarget

Gets or sets the Connector or the ItemContainer (if AllowOnlyConnectors is false) that we're previewing.

```
public object? PreviewTarget { get; set; }
```

Property Value

object

Source

Gets or sets the Connector's DataContext that started this pending connection.

```
public object? Source { get; set; }
```

Property Value

object

SourceAnchor

Gets or sets the starting point for the connection.

```
public Point SourceAnchor { get; set; }
```

Property Value

Point

StartedCommand

Gets or sets the command to invoke when the pending connection is started. Will not be invoked if ConnectionStartedCommand is used. Source will be set to the Connector's DataContext that started this connection and will also be the command's parameter.

```
public ICommand? StartedCommand { get; set; }
```

Property Value

ICommand

Stroke

Gets or sets the stroke color of the connection.

```
public Brush Stroke { get; set; }
```

Property Value

Brush

StrokeDashArray

Gets or sets the pattern of dashes and gaps that is used to outline the connection.

```
public DoubleCollection StrokeDashArray { get; set; }
```

Property Value

DoubleCollection

StrokeThickness

Gets or set the connection thickness.

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

Property Value

double

Target

Gets or sets the Connector's DataContext (or potentially an ItemContainer's DataContext if AllowOnly Connectors is false) that the Source can connect to. Only set when the connection is completed (see CompletedCommand).

```
public object? Target { get; set; }
```

Property Value

object

TargetAnchor

Gets or sets the end point for the connection.

```
public Point TargetAnchor { get; set; }
```

Property Value

Point

Methods

GetIsOverElement(UIElement)

```
public static bool GetIsOverElement(UIElement elem)
```

Parameters

`elem` UIElement

Returns

bool

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

OnPendingConnectionCompleted(object, PendingConnectionEventArgs)

```
protected virtual void OnPendingConnectionCompleted(object sender,
PendingConnectionEventArgs e)
```

Parameters

`sender` object

`e` PendingConnectionEventArgs

OnPendingConnectionDrag(object, PendingConnectionEventArgs)

```
protected virtual void OnPendingConnectionDrag(object sender, PendingConnectionEventArgs e)
```

Parameters

`sender` object

`e` PendingConnectionEventArgs

OnPendingConnectionStarted(object, PendingConnectionEventArgs)

```
protected virtual void OnPendingConnectionStarted(object sender,
PendingConnectionEventArgs e)
```

Parameters

`sender` object

`e` PendingConnectionEventArgs

SetIsOverElement(UIElement, bool)

```
public static void SetIsOverElement(UIElement elem, bool value)
```

Parameters

`elem` UIElement

`value` bool

Class PendingConnectionEventArgs

Namespace: Nodify

Assembly: Nodify.dll

Provides data for PendingConnection related routed events.

```
public class PendingConnectionEventArgs : RoutedEventArgs
```

Inheritance

object ← EventArgs ← RoutedEventArgs ← PendingConnectionEventArgs

Constructors

PendingConnectionEventArgs(object)

Initializes a new instance of the PendingConnectionEventArgs class using the specified SourceConnector.

```
public PendingConnectionEventArgs(object sourceConnector)
```

Parameters

`sourceConnector` object

The DataContext of a related Connector.

Properties

Anchor

Gets or sets the Anchor of the Connector that raised this event.

```
public Point Anchor { get; set; }
```

Property Value

Point

Canceled

Gets or sets a value that indicates whether this PendingConnection was cancelled.

```
public bool Canceled { get; set; }
```

Property Value

bool

OffsetX

Gets or sets the distance from the SourceConnector in the X axis.

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

Property Value

double

OffsetY

Gets or sets the distance from the SourceConnector in the Y axis.

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

Property Value

double

SourceConnector

Gets the DataContext of the Connector that started this PendingConnection.

```
public object SourceConnector { get; }
```

Property Value

object

TargetConnector

Gets or sets the DataContext of the target Connector when the PendingConnection is completed.

```
public object? TargetConnector { get; set; }
```

Property Value

object

Methods

InvokeEventHandler(Delegate, object)

When overridden in a derived class, provides a way to invoke event handlers in a type-specific way, which can increase efficiency over the base implementation.

```
protected override void InvokeEventHandler(Delegate genericHandler, object genericTarget)
```

Parameters

`genericHandler` Delegate

The generic handler / delegate implementation to be invoked.

`genericTarget` object

The target on which the provided handler should be invoked.

Delegate PendingConnectionEventHandler

Namespace: Nodify

Assembly: Nodify.dll

Represents the method that will handle PendingConnection related routed events.

```
public delegate void PendingConnectionEventHandler(object sender,
PendingConnectionEventArgs e)
```

Parameters

`sender` object

The object where the event handler is attached.

`e` PendingConnectionEventArgs

The event data.

Delegate PreviewLocationChanged

Namespace: Nodify

Assembly: Nodify.dll

Delegate used to notify when an ItemContainer is previewing a new location.

```
public delegate void PreviewLocationChanged(Point newLocation)
```

Parameters

`newLocation` Point

The new location.

Class ResizeEventArgs

Namespace: Nodify

Assembly: Nodify.dll

Provides data for resize related routed events.

```
public class ResizeEventArgs : RoutedEventArgs
```

Inheritance

object ← EventArgs ← RoutedEventArgs ← ResizeEventArgs

Constructors

ResizeEventArgs(Size, Size)

Initializes a new instance of the ResizeEventArgs class with the previous and the new Size.

```
public ResizeEventArgs(Size previousSize, Size newSize)
```

Parameters

`previousSize` Size

The previous size associated with this event.

`newSize` Size

The new size associated with this event.

Properties

NewSize

Gets the new size of the object.

```
public Size NewSize { get; }
```

Property Value

Size

PreviousSize

Gets the previous size of the object.

```
public Size PreviousSize { get; }
```

Property Value

Size

Methods

InvokeEventHandler(Delegate, object)

When overridden in a derived class, provides a way to invoke event handlers in a type-specific way, which can increase efficiency over the base implementation.

```
protected override void InvokeEventHandler(Delegate genericHandler, object genericTarget)
```

Parameters

`genericHandler` Delegate

The generic handler / delegate implementation to be invoked.

`genericTarget` object

The target on which the provided handler should be invoked.

Delegate ResizeEventHandler

Namespace: Nodify

Assembly: Nodify.dll

Represents the method that will handle resize related routed events.

```
public delegate void ResizeEventHandler(object sender, ResizeEventArgs e)
```

Parameters

`sender` object

The sender of this event.

`e` ResizeEventArgs

The event data.

Class SelectionHelper

Namespace: Nodify

Assembly: Nodify.dll

Helps with selecting ItemContainers and updating the SelectedArea and IsSelecting properties.

```
public sealed class SelectionHelper
```

Inheritance

object ← SelectionHelper

Constructors

SelectionHelper(NodifyEditor)

Constructs a new instance of a SelectionHelper.

```
public SelectionHelper(NodifyEditor host)
```

Parameters

`host` NodifyEditor

The editor to select items from.

Methods

Abort()

Aborts the current selection.

```
public void Abort()
```

End()

Commits the current selection to the editor.

```
public void End()
```

Start(Point, SelectionType)

Attempts to start a new selection.

```
public void Start(Point location, SelectionHelper.SelectionType selectionType)
```

Parameters

`location` Point

The location inside the graph.

`selectionType` SelectionHelper.SelectionType

The type of selection.

Remarks

Will not do anything if selection is in progress.

Update(Point)

Update the end location for the selection.

```
public void Update(Point endLocation)
```

Parameters

`endLocation` Point

An absolute location.

Enum SelectionHelper.SelectionType

Namespace: Nodify

Assembly: Nodify.dll

Available selection logic.

```
public enum SelectionHelper.SelectionType
```

Fields

`Append = 2`

Adds items to the current selection.

`Invert = 3`

Inverts the selection.

`Remove = 1`

Removes items from existing selection.

`Replace = 0`

Replaces the old selection.

Class StateNode

Namespace: Nodify

Assembly: Nodify.dll

Represents a control that acts as a Connector.

```
[TemplatePart(Name = "PART_Content", Type = typeof(UIElement))]
public class StateNode : Connector
```

Inheritance

object ← DispatcherObject ← DependencyObject ← Visual ← UIElement ← FrameworkElement ← Control ← Connector ← StateNode

Inherited Members

Connector.ElementConnector , Connector.PendingConnectionStartedEvent , Connector.PendingConnectionCompletedEvent , Connector.PendingConnectionDragEvent , Connector.DisconnectEvent , Connector.PendingConnectionStarted , Connector.PendingConnectionCompleted , Connector.PendingConnectionDrag , Connector.Disconnect , Connector.AnchorProperty , Connector.IsConnectedProperty , Connector.DisconnectCommandProperty , Connector.IsPendingConnectionProperty , Connector.Anchor , Connector.IsConnected , Connector.IsPendingConnection , Connector.DisconnectCommand , Connector.Thumb , Connector.Container , Connector.Editor , Connector.OptimizeSafeZone , Connector.OptimizeMinimumSelectedItems , Connector.EnableOptimizations , Connector.AllowPendingConnectionCancellation , Connector.EnableStickyConnections , Connector.OnRenderSizeChanged(SizeChangedInfo) , Connector.UpdateAnchorOptimized(Point) , Connector.UpdateAnchor(Point) , Connector.UpdateAnchor() , Connector.OnLostMouseCapture(MouseEventArgs) , Connector.OnKeyUp(KeyEventArgs) , Connector.OnMouseMove(MouseEventArgs) , Connector.OnConnectorDrag(Vector) , Connector.OnConnectorDragStarted() , Connector.OnConnectorDragCompleted(bool) , Connector.OnDisconnect()

Fields

ContentProperty

```
public static readonly DependencyProperty ContentProperty
```

Field Value

DependencyProperty

ContentTemplateProperty

```
public static readonly DependencyProperty ContentTemplateProperty
```

Field Value

DependencyProperty

CornerRadiusProperty

```
public static readonly DependencyProperty CornerRadiusProperty
```

Field Value

DependencyProperty

ElementContent

```
protected const string ElementContent = "PART_Content"
```

Field Value

string

HighlightBrushProperty

```
public static readonly DependencyProperty HighlightBrushProperty
```

Field Value

DependencyProperty

Properties

Content

Gets or sets the data for the control's content.

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

Property Value

object

ContentControl

Gets the ContentControl control of this StateNode.

```
protected UIElement? ContentControl { get; }
```

Property Value

UIElement

ContentTemplate

Gets or sets the template used to display the content of the control's header.

```
public DataTemplate ContentTemplate { get; set; }
```

Property Value

DataTemplate

CornerRadius

Gets or sets a value that represents the degree to which the corners of the StateNode are rounded.

```
public CornerRadius CornerRadius { get; set; }
```

Property Value

CornerRadius

HighlightBrush

Gets or sets the brush used when the IsOverElementProperty attached property is true for this StateNode.

```
public Brush HighlightBrush { get; set; }
```

Property Value

Brush

Methods

OnApplyTemplate()

When overridden in a derived class, is invoked whenever application code or internal processes call ApplyTemplate().

```
public override void OnApplyTemplate()
```

OnMouseDown(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseDown attached event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseDown(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. This event data reports details about the mouse button that was pressed and the handled state.

OnMouseUp(MouseButtonEventArgs)

Invoked when an unhandled System.Windows.Input.Mouse.MouseUp routed event reaches an element in its route that is derived from this class. Implement this method to add class handling for this event.

```
protected override void OnMouseUp(MouseButtonEventArgs e)
```

Parameters

e MouseButtonEventArgs

The MouseButtonEventArgs that contains the event data. The event data reports that the mouse button was released.