MiscDiagramTree

Inherits From:	MiscLayoutTree
Declared In:	MiscDiagramTree.h

Class Description

The MiscDiagramTree class extends the tree data structure represented by the **MiscLayoutTree** class with drawing capabilities.   An instance of this class can draw itself in a **MiscTreeDiagramView** instance.   It is defined by a shape (an instance of **MiscDiagramShape**), a label and an image.   Its appearance is also formed by two defining styles: its own node style (an instance of **MiscNodeStyle**) and the style of the tree diagram it belongs to (an instance of **MiscTreeStyle**).

paste.eps ¬

The drawing has been split into several methods for efficiency reasons.   It enables the drawing of all shadows, then all fills then all outlines...   etc.   This minimizes the changes that have to be set in the postscript drawing state.

In addition to the basic drawing there's drawing functionality for visual feedback on the state of the MiscDiagramTree instance, i.e.   if it's selected, collapsed, has attachments and so on.   Also there are methods for interactive drawing when the instance is in a resizing event in a view.

Finding out the bounding rectangle of an MiscDiagramTree instance including the selection handles, the attachments handles, considering the linewidth and so on...   is done with the methods **nodeDrawBounds** and **branchDrawBounds**.

All MiscDiagramTree instances of a tree are part of the same diagram, an instance of **MiscTreeDiagram**. The method **belongsToDiagram** returns the diagram of which the receiver is part of.   The owning diagram can be set with **setOwningDiagram:**.

Instance Variables

No @public or @protected instance variables declared in this class.

Method Types

Allocating and initializing an MiscDiagramTree instance	
	+ tree
	+ treeWithLabel:shapeType:
	+ treeWithSize:shapeType:
	- initWithLabel:shapeType:
	- initWithSize:shapeType:
Drawing	- drawShadow:
	- drawOutline:knobs:viewScale:
	- drawFill:
	- drawEnding:
	- drawCellsInView:attachments:
	- fillLines:andAddTo:
	- fillLines:andAddTo:ifInvolvedInRect:
Accessing geometric data	- nodeDrawBounds
	- branchDrawBounds
	- textBounds
	- endPoint
	- parentEndPoint
	- childEndPoint
Accessing and manipulating shape	
	- shapeType
	- setShapeType:
	- shape

Accessing and manipulating node contents	
	- label
	- setLabel:
Managing node selection and hit detection	
	- isSelected
	- setSelected:
	- hit:controlPoint:outline:
	- controlPoint:
	- controlRect:
Setting owning diagram	- setDiagram:
	- diagram
Accessing and setting node style	
	- setNodeStyle:
	- nodeStyle
Interactive resizing in a view	- setResizingFlag
	- resetResizingFlag
	- drawResizingInView:viewScale:
	- drawResizing
	- resizeToControlPoint::

Class Methods

tree

+ **tree**

Creates, initializes and returns a MiscDiagramTree instance.

treeWithLabel:shapeType:

+ **treeWithLabel:**(NSAttributedString *)*aLabel*
  shapeType:(MiscShapeType)*aShapeType*

Creates, initializes and returns a MiscDiagramTree instance with label *aLabel* and shape type *aShapeType*.

treeWithSize:shapeType:

+ **treeWithSize:**(NSSize)*aSize*
  shapeType:(MiscShapeType)*aShapeType*

Creates, initializes and returns a MiscDiagramTree instance with size *aSize* and shape type *aShapeType*.

Instance Methods

branchDrawBounds

- (NSRect)**branchDrawBounds**

Returns the bounds of the tree rooted in the receiver taking into consideration style properties like linewidth and the extension of the various handles and signs like the selection handles, the attachments handles and the collapsed sign.

childEndPoint

- (NSPoint)**childEndPoint**

Returns the bending point on the receiver's side of the line connecting the receiver with its parent when the line type is *MiscBendLineType* (see the **MiscTreeStyle** class).   In this implementation it's calculated from the *bendingFactor* and the *distanceToParent* parameters (see the **MiscTreeStyle** and **MiscLayoutTree** classes). You can overwrite this method to provide some other bending point.

controlPoint:

- (NSPoint)**controlPoint:**(int)*aInt*

Returns the coordinates of the selection knob or control point *aInt*.   There are four selection knobs positioned in the four corners of the bounding rectangle and numbered counterclockwise starting with 0.

controlRect:

- (NSRect)**controlRect:**(int)*aInt*

Returns the rectangle enclosing the selection knob or control point *aInt*. There are four selection knobs positioned in the four corners of the bounding rectangle and numbered counterclockwise starting with 0.

diagram

- (MiscTreeDiagram *)**diagram**

Returns the diagram the receiver belongs to.

drawCellsInView:attachments:

- (void)**drawCellsInView:**(NSView *)*aView*
  attachments:(BOOL)*aBool*

Draws the label of the receiver. There should be a view with locked focus when this method is called. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results. The method draws the attachments knobs if the receiver has attachments and *aBool* has the value YES. The method also draws the collapsed sign if the receiveris collapsed and *aBool* has the value YES.

drawEnding:

- (void)**drawEnding:**(MiscDrawState *)*aState*

Draws the line endings of the receiver. There should be a view with locked focus when this method is called. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results. The method compares the setting it has to make to the postscript graphic state with the values in *aState*. *aState* should have valid values reflecting the current postscript graphic state. The changes to the postscript graphic state are written back in *aState*.

drawFill:

- (void)**drawFill:**(MiscDrawState *)*aState*

Draws the fill of the receiver. There should be a view with locked focus when this method is called. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results. The method compares the setting it has to make to the postscript graphic state with the values in *aState*. *aState* should have valid values reflecting the current postscript graphic state. The changes to the postscript graphic state are written back in *aState*.

drawOutline:knobs:viewScale:

- (void)**drawOutline:**(MiscDrawState *)*aState*
  knobs:(BOOL)*knobs*
  viewScale:(float)*viewScale*

Draws the outline of the receiver. There should be a view with locked focus when this method is called. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results. The method compares the setting it has to make to the postscript graphic state with the values in *aState*. *aState* should have valid values reflecting the current postscript graphic state. The changes to the postscript graphic state are written back in *aState*. The method draws the selection *knobs* if the receiver is selected and aBool has the value YES. The value of *viewScale* gives the scaling factor of the current postscript graphic state. The method calculates the size of the *knobs* such that it appears constant onscreen regardless of the scaling factor.

drawResizing

- (void)**drawResizing**

Draws the receiver in an interactive drawing mode.

drawResizingInView:viewScale:

- (void)**drawResizingInView:**(NSView *)*aView*
  viewScale:(float)*aViewScale*

Draws the receiver in an interactive drawing mode. Used for resizing in *aView*. The value of viewScale gives the scaling factor of the current postscript graphic state. The method calculates the size of the knobs such that it appears constant onscreen regardless of the scaling factor.

drawShadow:

- (void)**drawShadow:**(MiscDrawState *)*aState*

Draws the shadow of the receiver. There should be a view with locked focus and the shadow color already set when this method is called. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results.

endPoint

- (NSPoint)**endPoint**

Returns the endpoint on the receiver's side of the lines connecting the receiver with its parent and children. In this implementation it's the middle point of the receiver's shape. You can overwrite this method to provide some other endpoint.

fillLines:andAddTo:

- (void)**fillLines:**(MiscUserPath *)*linesPath*
  andAddTo:(NSMutableArray *)*nodes*

Extends the *linesPath* with the line forming the connection to the receiver's parent and adds the receiver to the *nodes* list for drawing. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results.

fillLines:andAddTo:ifInvolvedInRect:

- (void)**fillLines:**(MiscUserPath *)*linesPath*
  andAddTo:(NSMutableArray *)*nodes*
  ifInvolvedInRect:(NSRect)*aRect*

Extends the *linesPath* with the line forming the connection to the receiver's parent if the line intersects *aRect* and adds the receiver to the *nodes* list for drawing if the receiver's bounds intersect *aRect*. The method does not check if the receiver needs layout. Calling the method with a node which is not already layed out has unpredictable results.

hit:controlPoint:outline:

- (BOOL)**hit:**(MiscHitPath *)*hitPath*
  controlPoint:(int *)*aInt*
  outline:(BOOL *)*aBOOL*

Returns YES if *hitPath* intersects one of the selection knobs.   In this case *aInt* holds the number of the selection knob.   There are four selection knobs positioned in the four corners of the bounding rectangle and numbered counterclockwise starting with 0.   If *hitPath* does not intersect a knob then returns YES if the receiver's shape intersects *hitPath* and NO if not.   In this case *aInt* is set to -1.   If *hitPath* intersect the outline *aBOOL* is set to YES.

initWithLabel:shapeType:

- **initWithLabel:**(NSAttributedString *)*aLabel*
  shapeType:(MiscShapeType)*aShapeType*

Initializes a newly created instance of MiscDiagramTree with label *aLabel* and shape type *aShapeType*.   It calculates the size to fit the label.

initWithSize:shapeType:

- **initWithSize:**(NSSize)*aSize*
  shapeType:(MiscShapeType)*aShapeType*

Initializes a newly created instance of MiscDiagramTree with size *aSize* and shape type *aShapeType*.   This is the designated initializer of this class.

isSelected

- (BOOL)**isSelected**

Returns whether the receiver is selected.

label

- (NSAttributedString *)**label**

Returns the receiver's label.

nodeDrawBounds

- (NSRect)**nodeDrawBounds**

Returns the bounds of the receiver taking into consideration style properties like linewidth and the extension of the various handles and signs like the selection handles, the attachments handles and the collapsed sign.

nodeStyle

- (MiscNodeStyle *)**nodeStyle**

Returns the node style of the receiver.

parentEndPoint

- (NSPoint)**parentEndPoint**

Returns the bending point on the receiver's side of the line connecting the receiver with its children when the line type is *MiscBendLineType* (see the **MiscTreeStyle** class).   In this implementation it's calculated from the *bendingFactor* and the *distanceToParent* parameters (see the **MiscTreeStyle** and **MiscLayoutTree** classes). You can overwrite this method to provide some other bending point.

resetResizingFlag

- (void)**resetResizingFlag**

Sets the drawing mode back to normal.

resizeToControlPoint::

- (void)**resizeToControlPoint:**(NSPoint)*aPos* **:**(int *)*aInt*

Temporarily resizes the receiver to reflect the new position *aPos* of selection knob *aInt*.   It also changes *aInt* to a new selection knob number if *aPos* "tumbles" the receiver.

setDiagram:

- (void)**setDiagram:**(MiscTreeDiagram *)*aDiagram*

Recursively sets the owning diagram of every node of the tree rooted in the receiver.

setLabel:

- (void)**setLabel:**(NSAttributedString *)*aLabel*

Sets the label of the receiver to *aLabel*. If the receiver has the *automatic resizing flag* set then the size changes to accomodate *aLabel*.

setNodeStyle:

- (void)**setNodeStyle:**(MiscNodeStyle *)*aStyle*

Sets the receiver's node style to *aStyle*.

setResizingFlag

- (void)**setResizingFlag**

Changes the drawing mode of the receiver to interactive drawing for resizing. In this mode only a light gray shadow of the receiver is drawn with **drawFill:**. The other drawing methods do nothing.

setSelected:

- (void)**setSelected:**(BOOL)*aBOOL*

Sets the receiver's selection status to *aBOOL*.

setShapeType:

- (void)**setShapeType:**(MiscShapeType)*aShapeType*

Sets the shape of the receiver to *aShapeType*. If the receiver has the *automatic resizing flag* set then the size of the receiver changes such that the shape inner bounds stay the same.

shape

- (MiscDiagramShape *)**shape**

Returns the shape of the receiver.

shapeType

- (MiscShapeType)**shapeType**

Returns the shape type of the receiver.

textBounds

- (NSRect)**textBounds**

Returns the text bounds of the receiver.   The label text of the receiver is clipped to this bounds.