MiscLayoutTree

Inherits From: NSObject
Declared In: MiscLayoutTree.h

Class Description

An instance of this class holds a tree data structure that can lay itself out. The following aesthetic rules are used:

Siblings of the same parent node lie along a line.
A subtree looks the same, regardless of where it occurs.
A tree is drawn as compact as possible.

A tree is layed out vertically or horizontally, depending on the layout type, which can be queried with **layoutType** and can be set to one of the values *MiscHorizontalTreeType* or *MiscVerticalTreeType* with **setLayoutType:**.

The layout distance between nodes is defined by two parameters: the distance to the parent *distanceToParent* and the border *border*. Both parameters remain the same for every node.

paste.eps ¬ 554503_paste.eps ¬

Every node in the tree has 3 pointers: a pointer to his parent, a pointer to a sibling and a pointer to his first child.

621169_paste.eps ¬

This way every node has a uniquely defined predecessor.

The position *pos* of the node is defined as the *pos* of the predecessor plus the *offset* of the node. Every node keeps track of the polygonial contour of its subtree. The layout algorithm makes sure the contours of siblings don't overlap. The algorithm was published by Sven Moen in IEEE Software, July 1990.

The MiscLayoutTree class provides methods for querying and setting the geometric parameters of the node. Setting the position of the node with **setPos:** is only allowed for the root node. The method **nodeBounds** returns a NSRect with the bounds of the node and the method **branchBounds** returns a NSRect with the bounds of the subtree rooted in the node.

BranchBounds.eps ¬

The methods **depthEnumerator** and **breadthEnumerator** permit sequential access of the elements of the tree, differing only in the way of travel through the elements (depth first or breadth first). The methods **addBranch:**, **insertBranch:at:**, **removeBranch:** and **removeBranchAt:** modify the tree data structure. A subtree can be collapsed into its root node with the method **collapse**.

All methods modifying the geometric parameters or the tree data structure call **invalidateLayout**. The method **layout** lays out the tree and marks the layout as valid.

Defined Types

SYNOPSIS

MiscPolyline

```
typedef struct _MiscPolyline{
        float dx,dy;
        struct _MiscPolyline *link;
} MiscPolyline;
```

DESCRIPTION

MiscPolyline is a linked list representing a polygon line.

SYNOPSIS

MiscPolygon

```
typedef struct _MiscPolygon{
        struct {
        MiscPolyline *head,*tail;
} lower,upper;
} MiscPolygon;
```

DESCRIPTION

MiscPolygon is made out of an upper (left) and a lower (right) polygon line.

SYNOPSIS

MiscLayoutTreeType

```
typedef enum _MiscLayoutTreeType{
        MiscHorizontalTreeType,
        MiscVerticalTreeType
} MiscLayoutTreeType;
```

DESCRIPTION

This types constants define the layout of the tree: horizontal or vertical.

SYNOPSIS

MiscLayoutTreeOrientation

```
typedef enum _MiscLayoutTreeOrientation{
        MiscStandardOrientation,
        MiscInverseOrientation
} MiscLayoutTreeOrientation;
```

DESCRIPTION

This types constants define the orientation of the tree (i.e. the position of the root node).

Instance Variables

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

Method Types

Allocating and initializing an MiscLayoutTree instance
+ tree
+ treeWithSize:
- initWithSize:
- copyNodeWithZone:

Laying out
- layout
- needsUpdateLayout
- invalidateLayout
- layoutNode

Accessing and manipulating geometric structure
- setPos:
- setSize:
- setDistanceToParent:
- setBorder:
- pos
- offset
- size
- distanceToParent
- border
- width
- height
- nodeBounds
- branchBounds
- layoutType
- setLayoutType:
- orientation
- setOrientation:

Accessing and manipulating tree structure
- sibling
- child
- parent
- depthEnumerator
- breadthEnumerator
- branchAt:
- numberOfChildren
- addBranch:
- insertBranch:at:
- removeBranch:
- removeBranchAt:
- moveBranchAt:to:
- isCollapsed
- collapse
- expand
- toggleCollapse

Accessing and manipulating node contents
- setTag:
- tag

Interactive resizing in a view
- setTempNodeBounds:

Class Methods

tree

+ **tree**

Creates, initializes and returns a MiscLayoutTree instance.

treeWithSize:

+ **treeWithSize:**(NSSize)*aSize*

Creates, initializes and returns a MiscLayoutTree instance with size *aSize*.

Instance Methods

addBranch:

- (void)**addBranch:**(MiscLayoutTree *)*branch*

Adds the *branch* to the receiver's list of children. Marks the layout as invalid with **invalidateLayout**.

border

- (float)**border**

Returns the border property of the tree the receiver belongs to.

branchAt:

- (MiscLayoutTree *)**branchAt:**(unsigned)*index*

Returns the child numbered *index* (starting at zero) in the list of children of the receiver.

branchBounds

- (NSRect)**branchBounds**

Returns the bounds of the subtree rooted in the receiver.

breadthEnumerator

- (NSEnumerator *)**breadthEnumerator**

Returns an enumerator object that lets you access each node in the tree, starting with the root node and traversing the tree breadth first.

child

- (MiscLayoutTree *)**child**

Returns the first child of the receiver.

collapse

- (void)**collapse**

Collapses the subtree rooted in the receiver and marks the layout as invalid with **invalidateLayout**. Does nothing if subtree is already collapsed.

copyNodeWithZone:

- **copyNodeWithZone:**(NSZone *)*zone*

This method is called by **copy** and by **copyWithZone:** to copy the node specific instance variables. You should overwrite this method rather than **copyWithZone:** in a subclass. The reason is the method **copyWithZone:** makes a deep copy of the subtree rooted in the receiver, thus must take care of the tree structure.

depthEnumerator

- (NSEnumerator *)**depthEnumerator**

Returns an enumerator object that lets you access each node in the tree, starting with the root node and traversing the tree depth first.

distanceToParent

- (float)**distanceToParent**

Returns the distance to parent property of the tree the receiver belongs to.

expand

- (void)**expand**

Expands the collapsed subtree rooted in the receiver and marks the layout as invalid with **invalidateLayout**. Does nothing if subtree is already expanded.

height

- (float)**height**

Returns the height of the receiver.

initWithSize:

- **initWithSize:**(NSSize)*aSize*

Initializes a newly created instance of MiscLayoutTree with *aSize*. This is the designated initializer of this class.

insertBranch:at:

- (void)**insertBranch:**(MiscLayoutTree *)*branch*
  at:(unsigned)*index*

Inserts the *branch* in the receiver's list of children at position *index*. Marks the layout as invalid with **invalidateLayout**.

invalidateLayout

- (void)**invalidateLayout**

Marks the layout of the receiver as invalid. The layout of the receiver's parent will also be invalid and so on up to the root node of the tree the receiver belongs to. If you overwrite this method make *[**super** invalidateLayout]* the first statement of your methods body.

isCollapsed

- (BOOL)**isCollapsed**

Returns whether receiver is a collapsed subtree.

layout

- (void)**layout**

Lays out the tree and resets updateLayout to NO. A collapsed subtree is layed out as a leaf. This method can only be called on receivers that are root nodes of MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Don't overwrite this method in subclasses. Overwrite instead **layoutNode** for node specific layouts.

layoutNode

- (void)**layoutNode**

This method should be overwritten by subclasses to finish the layout for nodes. When this method is called the receiver's layout position has already been calculated. You should not call this method directly. When you overwrite don't forget to start with a *[**super** layoutNode]*.

layoutType

- (MiscLayoutTreeType)**layoutType**

Returns the type of the tree (MiscHorizontalTreeType for horizontal layout or MiscVerticalTreeType vertical for vertical layout).

moveBranchAt:to:

- (void)**moveBranchAt:**(unsigned)*index* **to:**(unsigned)*otherIndex*

Changes position of branch in the receiver's list of children from *index* to *otherIndex*. Marks the layout as invalid with **invalidateLayout**.

needsUpdateLayout

- (BOOL)**needsUpdateLayout**

Returns whether the receiver needs to update the layout. This method can only be called on receivers that are root nodes of MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Updating is done by sending the receiver the **layout** message.

nodeBounds

- (NSRect)**nodeBounds**

Returns the node bounds of the receiver.

numberOfChildren

- (unsigned)**numberOfChildren**

Returns the number of children of the receiver.

offset

- (NSPoint)**offset**

Returns the offset of the receiver relative to its predecessor.

orientation

- (MiscLayoutTreeOrientation)**orientation**

Returns the orientation of the tree.

parent

- (MiscLayoutTree *)**parent**

Returns the parent of the receiver.

pos

- (NSPoint)**pos**

Returns the position of the receiver.

removeBranch:

- (void)**removeBranch:**(MiscLayoutTree *)*branch*

Removes *branch* from the receiver's list of children. Marks the layout as invalid with **invalidateLayout**.

removeBranchAt:

- (void)**removeBranchAt:**(unsigned)*index*

Removes the branch at position *index* in the receiver's list of children. Marks the layout as invalid with **invalidateLayout**.

setBorder:

- (void)**setBorder:**(float)*aBorder*

Sets the border of the receiver to aDistance. The border is a property concerning the whole tree, so this method can only be called on receivers that are root nodes of a MiscLayoutTree, i.e. nodes with the parent

pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Marks the layout as invalid with **invalidateLayout**.

setDistanceToParent:

- (void)**setDistanceToParent:**(float)*aDistance*

Sets the distance to parent of the receiver to *aDistance*. The distance to the parent node is a property concerning the whole tree, so this method can only be called on receivers that are root nodes of a MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Marks the layout as invalid with **invalidateLayout**.

setLayoutType:

- (void)**setLayoutType:**(MiscLayoutTreeType)*aType*

Sets the layout type of the receiver to *aType*. The layout type is a property concerning the whole tree, so this method can only be called on receivers that are root nodes of a MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Marks the layout as invalid with **invalidateLayout**.

setOrientation:

- (void)**setOrientation:**(MiscLayoutTreeOrientation)*anOrientation*

Sets the orientation of the receiver to aType. The orientation is a property concerning the whole tree, so this method can only be called on receivers that are root nodes of a MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. Marks the layout as invalid with **invalidateLayout**. Note: Has no effect in this version (i.e. layout is independent of this parameter). This is due to a restriction in the layout algorithm which requires a certain monotony property in the polylines.

setPos:

- (void)**setPos:**(NSPoint)*aPos*

Sets the position of the receiver to *aPos*. This method can only be called on receivers that are root nodes of MiscLayoutTree, i.e. nodes with the parent pointer **nil**. Calling this method with other nodes raises a **NSGenericException**. The position of a child node is calculated from the position of the predecessor node and the offset of the child node.

setSize:

- (void)**setSize:**(NSSize)*aSize*

Sets the size of the receiver to *aSize* and marks the layout as invalid with **invalidateLayout**.

setTag:

- (void)**setTag:**(int)*aTag*

Provides a way to identify nodes.

setTempNodeBounds:

- (void)**setTempNodeBounds:**(NSRect)*newBounds*

Sets the bounds of the receiver to *newBounds* overriding the bounds calculated by the layout algorithm. It does not invalidate the layout from before the method is called. This method is used for interactive defining the size of a node in the MiscTreeDiagram framework.

sibling

- (MiscLayoutTree *)**sibling**

Returns the sibling of the receiver.

size

- (NSSize)**size**

Returns the size of the receiver.

tag

- (int)**tag**

Returns the tag of the node.

toggleCollapse

- (void)**toggleCollapse**

Expands or collapses the subtree rooted in the receiver according to its current state and marks the layout as invalid with **invalidateLayout**.

width

- (float)**width**

Returns the width of the receiver.