

DB Tree Table View

Return Types

[Binding](#) [Column](#)

Property Summary

[String](#) [#bgcolor](#)

[Boolean](#) [#enabled](#)

[String](#) [#fgcolor](#)

[Object](#)[] [#selectionPath](#)
Get/Set the selection (path), array with pk records values (only single pk key supported)

[String](#) [#toolTipText](#)

[Boolean](#) [#transparent](#)

[Boolean](#) [#visible](#)

Method Summary

Number	#addRoots (foundset) Add foundset to the list of foundsets used to create the tree's root nodes.
Binding	#createBinding (datasource/servername, [tablename]) Create and add binding object for a database table used to set data bindings for nodes.
Column	#createColumn (servername, tablename, header, fieldname, [preferredWidth]) Create and add new column to the tree table
RelationInfo	#createRelationInfo () Create relation info object used to set multiple child relations for a tree node
Number	#getHeight () Returns the height of the tree.
Number	#getLocationX () Get the x coordinate of the location of the tree.
Number	#getLocationY () Get the y coordinate of the location of the tree.
String	#getName () Returns the name of the tree.
Number	#getWidth () Returns the width of the tree.
Boolean	#isNodeExpanded (path) Check the path (array with pk records values (only single pk key supported)) expanded status
void	#refresh () Refresh the tree display
void	#removeAllColumns () Remove all columns but the tree column from the tree table
void	#removeAllRoots () Remove all root foundsets
void	#setExpandNode (path, expand_collapse) Expand/collapse the path, array with pk records values (only single pk key supported)
void	#setFont (spec) Sets the specified font as the font of the tree.
void	#setLocation (x, y) Sets the location of the tree.
void	#setNodeLevelVisible (level, visible) Set the level of visible nodes (expand or collapse to certain level)
void	#setOnDrag (callback) Set method to be called when a drag is started on the tree.
void	#setOnDragEnd (callback) Set method to be called when a drag of on the tree is ended.
void	#setOnDragOver (callback) Set method to be called during a drag over the tree.
void	#setOnDrop (callback) Set method to be called on a drop on the tree.
void	#setRowHeight (rowHeight) Set row height
void	#setSize (x, y) Sets the size of the tree.
void	#setTreeColumnHeader (treeColumnHeader) Set the header text for the tree column
void	#setTreeColumnPreferredWidth (treeColumnPreferredWidth) Set the preferred width in pixels for the tree column

Property Details

bgcolor

Replace with description

Returns

[String](#)

enabled

Replace with description

Returns

[Boolean](#)

fgcolor

Replace with description

Returns

[String](#)

selectionPath

Get/Set the selection (path), array with pk records values (only single pk key supported)

Returns

[Object\[\]](#)

Sample

```
elements.customer_id.selectionPath = new Array(14,24,45,67);  
var currentSelectionArray = elements.customer_id.selectionPath;
```

toolTipText

Replace with description

Returns

[String](#)

transparent

Replace with description

Returns

[Boolean](#)

visible

Replace with description

Returns

[Boolean](#)

Method Details

addRoots

[Number](#) **addRoots**(foundset)

Add foundset to the list of foundsets used to create the tree's root nodes.

Returns the number of added root nodes.

Note: the bean will use a clone of the foundset, so any changes on the foundset parameter will be ignored in the tree.

Parameters

foundset

Returns

[Number](#)

Sample

```
//Add foundset to the list of foundsets used to create the tree's root nodes.  
Returns the number of added root nodes.  
Note: the bean will use a clone of the foundset, so any changes on the foundset parameter will be ignored in  
the tree.  
var addedRootNodes = elements.customer_id.addRoots(foundset);
```

createBinding

[Binding](#) **createBinding**(datasource/servername, [tablename])

Create and add binding object for a database table used to set data bindings for nodes.

Parameters

datasource/servername

[tablename]

Returns

[Binding](#)

Sample

```
var companies_binding = elements.customer_id.createBinding('example_data', 'companies');
companies_binding.setTextDataProvider('company_name');
companies_binding.setNRelationName('companies_to_companies');
companies_binding.setImageURLDataProvider('type_icon');
companies_binding.setChildSortDataProvider('company_sort');
```

createColumn

Column **createColumn**(servername, tablename, header, fieldname, [preferredWidth])

Create and add new column to the tree table

Parameters

servername

tablename

header

fieldname

[preferredWidth]

Returns

Column

Sample

```
elements.customer_id.createColumn('servername', 'tablename', 'header text', 'tablefieldname', 150);
```

createRelationInfo

RelationInfo **createRelationInfo**()

Create relation info object used to set multiple child relations for a tree node

Returns

RelationInfo

Sample

```
var company_relations = new Array();
company_relations[0] = elements.customer_id.createRelationInfo();
company_relations[0].setLabel('Employees');
company_relations[0].setNRelationName('companies_to_employees');
company_relations[1] = elements.customer_id.createRelationInfo();
company_relations[1].setLabel('Customers');
company_relations[1].setNRelationName('companies_to_customers');
companies_binding.setNRelationInfos(company_relations);
```

getHeight

Number **getHeight**()

Returns the height of the tree.

Returns

Number

Sample

```
elements.customer_id.getHeight();
```

getLocationX

Number **getLocationX**()

Get the x coordinate of the location of the tree.

Returns

Number

Sample

```
elements.customer_id.getLocationX();
```

getLocationY

Number **getLocationY**()

Get the y coordinate of the location of the tree.

Returns

Number

Sample

```
elements.customer_id.getLocationY();
```

getName

String **getName()**

Returns the name of the tree.

Returns

String

Sample

```
elements.customer_id.getName();
```

getWidth

Number **getWidth()**

Returns the width of the tree.

Returns

Number

Sample

```
elements.customer_id.getWidth();
```

isNodeExpanded

Boolean **isNodeExpanded(path)**

Check the path (array with pk records values (only single pk key supported)) expanded status

Parameters

path

Returns

Boolean

Sample

```
var pathArray = new Array(14,24,45,67);
elements.customer_id.isNodeExpanded(pathArray);
```

refresh

void **refresh()**

Refresh the tree display

Returns

void

Sample

```
elements.customer_id.refresh();
```

removeAllColumns

void **removeAllColumns()**

Remove all columns but the tree column from the tree table

Returns

void

Sample

```
elements.customer_id.removeAllColumns();
```

removeAllRoots

void **removeAllRoots()**

Remove all root foundsets

Returns

void

Sample

```
elements.customer_id.removeAllRoots();
```

setExpandNode

void **setExpandNode**(path, expand_collapse)

Expand/collapse the path, array with pk records values (only single pk key supported)

Parameters

path

expand_collapse

Returns

void

Sample

```
var pathArray = new Array(14,24,45,67);  
elements.customer_id.setExpandNode(pathArray, true);
```

setFont

void **setFont**(spec)

Sets the specified font as the font of the tree.

Parameters

spec

Returns

void

Sample

```
elements.customer_id.setFont('Times New Roman, 1, 22');
```

setLocation

void **setLocation**(x, y)

Sets the location of the tree.

Parameters

x

y

Returns

void

Sample

```
elements.customer_id.setLocation(120,80);
```

setNodeLevelVisible

void **setNodeLevelVisible**(level, visible)

Set the level of visible nodes (expand or collapse to certain level)

Parameters

level

visible

Returns

void

Sample

```
elements.customer_id.setNodeLevelVisible(globals.g_treeview_level, (globals.g_treeview_expand == 1 ? true :  
false));
```

setOnDrag

void **setOnDrag**(callback)

Set method to be called when a drag is started on the tree. For more details about the method arguments and return value check the same property of a form

Parameters

callback

Returns

void

Sample

```
elements.customer_id.setOnDrag(onDrag);
```

setOnDragEnd

void **setOnDragEnd**(callback)

Set method to be called when a drag of on the tree is ended. For more details about the method arguments and return value check the same property of a form

Parameters

callback

Returns

void

Sample

```
elements.customer_id.setOnDragEnd(onDragEnd);
```

setOnDragOver

void **setOnDragOver**(callback)

Set method to be called during a drag over the tree. For more details about the method arguments and return value check the same property of a form

Parameters

callback

Returns

void

Sample

```
elements.customer_id.setOnDragOver(onDragOver);
```

setOnDrop

void **setOnDrop**(callback)

Set method to be called on a drop on the tree. For more details about the method arguments and return value check the same property of a form

Parameters

callback

Returns

void

Sample

```
elements.customer_id.setOnDrop(onDrop);
```

setRowHeight

void **setRowHeight**(rowHeight)

Set row height

Parameters

rowHeight

Returns

void

Sample

```
elements.customer_id.setRowHeight(40);
```

setSize

void **setSize**(x, y)

Sets the size of the tree.

Parameters

x

y

Returns

void

Sample

```
elements.customer_id.setSize(400,300);
```

setTreeColumnHeader

void **setTreeColumnHeader**(treeColumnHeader)

Set the header text for the tree column

Parameters

treeColumnHeader

Returns

void

Sample

```
elements.customer_id.setTreeColumnHeader('Tree Column Header');
```

setTreeColumnPreferredWidth

void **setTreeColumnPreferredWidth**(treeColumnPreferredWidth)

Set the preferred width in pixels for the tree column

Parameters

treeColumnPreferredWidth

Returns

void

Sample

```
elements.customer_id.setTreeColumnPreferredWidth(200);
```