DROPICON

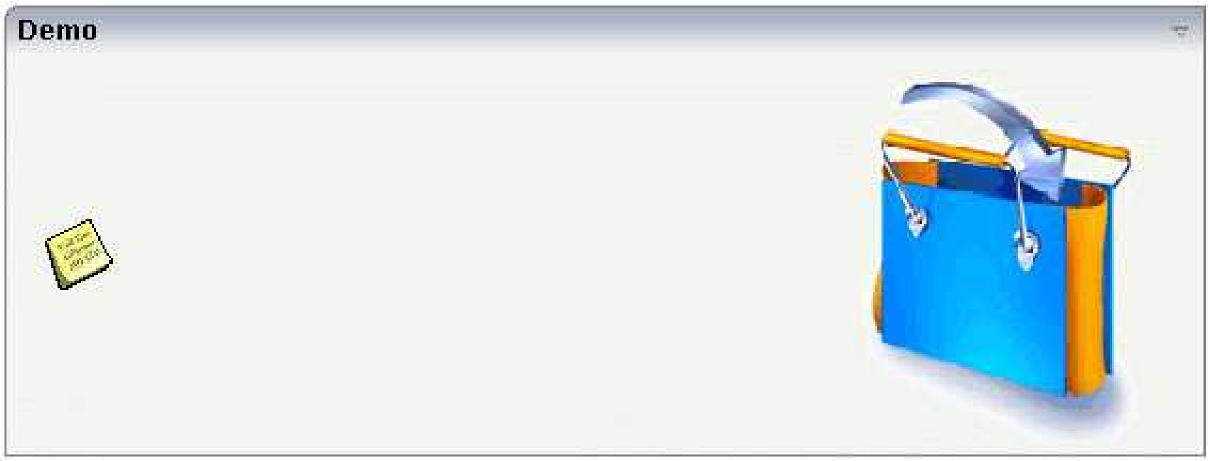
The DROPICON control is an icon that can be used in order to build drag-and-drop scenarios. A DROPICON can be defined as the starting point of a drag-and-drop operation or as the target point of a drag-and-drop operation.

The following topics are covered below:

- Example
- Dragging and Dropping Information from DROPICON to TREENODE3
- Dragging and Dropping Information from DROPICON to ICONLIST
- Properties

Example

Have a look at the following screen:



The user can click the left mouse button on the left icon (drag), move the mouse to the right icon and then release the mouse button (drop).

The configuration of drag and drop is quite simple: the icon that is used for starting drag-and-drop operations leaves a certain drag information - a plain string object. The receiving icon, on which the user performs the drop operation, receives both an event and the string object which was left by the icon from where the operation was started.

Have a look at the XML layout definition of the example above:

```
<rowarea name="Demo">
    <itr takefullwidth="true">
        <hdist width="10">
        </hdist>
        <dropicon image="../_DevelopersGuide/images/fav_notes.gif"
                  method="onSomeOtherFunctionality"
                  draginfo="Icon NOTICES has been dropped" dropmethod="onDrop"
                  dropinfoprop="onDropInfo">
        </dropicon>
        <hdist width="100%">
        </hdist>
        <dropicon image="../_DevelopersGuide/images/cishop.gif"
                  draginfo="HAND BAG has been dropped" dropmethod="onDrop"
                  dropinfoprop="onDropInfo">
        </dropicon>
        <hdist width="10">
        </hdist>
    </itr>
</rowarea>
```

In the left icon, the string that is used as drag information is defined using the `draginfo` property. (There is also a `draginfoprop` property that allows to specify this information using an adapter property instead of hard-wiring it in the layout.) You see that the icon still has a normal `method` property, i.e. it can still be used as a normal icon - invoking a certain adapter method when being chosen.

In the right icon, the `dropmethod` property defines the method that is called inside the adapter when the user drops certain information onto this icon. The property `dropinfoprop` defines the adapter property in which the string representing the drag information is written.

A DROPICON can be both defined as the starting point and as the target point of a drag-and-drop operation.

Dragging and Dropping Information from DROPICON to TREENODE3

The management of trees is described in *Working with Trees*. It is rcommended that you first read the general information about building trees with TREENODE3 there.

In a tree, each tree node is represented by a node object which is an extension of the `NODEInfo` class. If the user drops information from a DROPICON onto a tree node, then the method `reactOnDropGeneric` is called inside the node object. By calling the method `getDragInfo()`, you receive the drag information that was dropped onto the tree node.

Dragging and Dropping Information from DROPICON to ICONLIST

Each ICONLIST item is represented by an object of type `ICONLISTItem`. This object provides a method `reactOnDropGeneric()` which is called when information is dropped, and it provides a method `getDragInfo()` for accessing the dropped information.

Properties

Basic

image	URL that points to the image that is shown as icon. The URL either is an absolute URL or a relative URL. If using a relative URL then be aware of that the generated page is located directly inside your project's directory. Example: "images/icon.gif" points to an icon in an images-folder that is parallel to the page itself. "../HTMLBasedGUI/images/new.gif" point to a URL that is located inside a different project.	Obligatory	gif jpg jpeg
draginfo	String containing any kind of application data to identify the source DROPINFO control within a drag and drop process. Use property DROPINFOPROP to return this data on runtime.	Optional	
draginfoprop	Name of an adapter property that provides for information that is passed to the adapter when dropping this control over another DROPICON. Do not use this property (or property DROPINFO respectively) if you do not want the user to drag this control. The server side property needs to be of type "String".	Optional	
dropinfoprop	Name of an adapter property to that the "drag info" of the dragged DROPICON control is set. Do not use this property if this control should not accept other DROPICON controls within a drag and drop process (i.e. is not a drop target). The server side property needs to be of type "String".	Optional	
dropmethod	Method of your adapter object that is executed when the user is dragging another DROPICON control over this control and drops it there. Do not use this attribute if this control should not accept other DROPICON controls within a drag and drop process (i.e. is not a drop target).	Sometimes obligatory	
method	Method of your adapter object that is executed when clicking on the control.	Sometimes obligatory	
comment	Comment without any effect on rendering and behaviour. The comment is shown in the layout editor's tree view.	Optional	
Binding			
draginfoprop	(already explained above)		
dropinfoprop	(already explained above)		
dropmethod	(already explained above)		

imageprop	Name of adapter property that provides as value the URL of the image that is shown inside the control. The URL must either be an absolute URL or a relative URL.	Optional	
method	(already explained above)		
visibleprop	Name of an adapter property that provides the information if this control is displayed or not. As consequence you can control the visibility of the control dynamically. The server side property needs to be of type "boolean".	Optional	
titleprop	Property of adapter that dynamically defines the title of the control. The title is displayed as tool tip when ther user moves the mouse onto the control.	Optional	
Appearance			
image	(already explained above)		
invisiblemode	If the visibility of the control is determined dynamically by an adapter property then there are two rendering modes if the visibility is "false": (1) "invisible": the control is not visible. (2) "disabled": the control is deactivated: it is "grayed" and does not show any roll over effects any more.	Optional	invisible cleared
imageinactive	If the visibility is dynamically controlled by using the INVISIBLEPROP then there are two ways the icon reacts if the corresponding property passes back "false". If you want the icon to switch into an inactive status then define inside this property the URL of the image that is the inactive counter part to the normal icon image. Maybe the image is a grayed version of the normal icon image. If you do not define a value for this property then the icon is made invisible.	Optional	
imagewidth	Pixel width of the image that is shown inside the icon. If not defined then the icon is rendered with its normal width.	Optional	
imageheight	Pixel height of the image that is shown inside the icon. If not defined then the icon is rendered with its normal height.	Optional	

withdistance	If set to "true" then 2 pixels of distance are kept on the left and on the right of the icon. Reason behing: if arranging several icons inside one table row (ITR, TR) then a certain distance is kept between the icons when this property is set to "true".	Optional	true false
align	Horizontal alignment of control in its column. Each control is "packaged" into a column. The column itself is part of a row (e.g. ITR or TR). Sometimes the size of the column is bigger than the size of the control itself. In this case the "align" property specifies the position of the control inside the column. In most cases you do not require the align control to be explicitly defined because the size of the column around the controls exactly is sized in the same way as the contained control. If you want to directly control the alignment of text: in most text based controls there is an explicit property "textalign" in which you align the control’s contained text.	Optional	left center right
valign	Vertical alignment of control in its column. Each control is "packaged" into a column. The column itself is part of a row (e.g. ITR or TR). Sometimtes the size of the column is bigger than the size of the control. In this case the "align" property specify the position of the control inside the column.	Optional	top middle bottom
colstyle	CSS style definition that is directly passed into this control. With the style you can individually influence the rendering of the control. You can specify any style sheet expressions. Examples are: border: 1px solid #FF0000 background-color: #808080 You can combine expressions by appending and separating them with a semicolon. Sometimes it is useful to have a look into the generated HTML code in order to know where direct style definitions are applied. Press right mouse-button in your browser and select the "View source" or "View frame’s source" function.	Optional	background-color: #FF0000 color: #0000FF font-weight: bold

spanstyle	CSS style definition that is directly passed into this control. With the style you can individually influence the rendering of the control. You can specify any style sheet expressions. Examples are: border: 1px solid #FF0000 background-color: #808080 You can combine expressions by appending and separating them with a semicolon. Sometimes it is useful to have a look into the generated HTML code in order to know where direct style definitions are applied. Press right mouse-button in your browser and select the "View source" or "View frame’s source" function.	Optional	background-color: #FF0000 color: #0000FF font-weight: bold
tabindex	Index that defines the tab order of the control. Controls are selected in increasing index order and in source order to resolve duplicates.	Optional	-1 0 1 2 5 10 32767
Online Help			
title	Text that is shown as tooltip for the control. Either specify the text "hard" by using this TITLE property - or use the TITLETEXTID in order to define a language dependent literal.	Optional	
titletextid	Text ID that is passed to the multi lanaguage management - representing the tooltip text that is used for the control.	Optional	
titleprop	(already explained above)		