

# COMBODYN2

The COMBODYN control is the dynamic counterpart of the COMBOFIX control. Whereas the selection options inside the COMBOFIX control are defined in a fixed way inside the page definition, the COMBODYN2 control offers the possibility to control the selection options dynamically in the application.

The following topics are covered below:

- Adapter Interface
- Properties

## Adapter Interface

```

DEFINE DATA PARAMETER
1 COSTCENTER (U) DYNAMIC
1 VALIDCOSTCENTERS (1:*)
2 ID (U) DYNAMIC
2 NAME (U) DYNAMIC
2 SELECTED (L)
END-DEFINE

```

## Properties

Basic			
valueprop	Name of the adapter parameter that provides the content of the control.	Obligatory	
validvaluesprop	Name of the adapter parameter that provides the valid values that are available as selectable options.	Obligatory	

width	<p>Width of the control.</p> <p>There are three possibilities to define the width:</p> <p>(A) You do not define a width at all. In this case the width of the control will either be a default width or - in case of container controls - it will follow the width that is occupied by its content.</p> <p>(B) Pixel sizing: just input a number value (e.g. "100").</p> <p>(C) Percentage sizing: input a percentage value (e.g. "50% "). Pay attention: percentage sizing will only bring up correct results if the parent element of the control properly defines a width this control can reference. If you specify this control to have a width of 50% then the parent element (e.g. an ITR-row) may itself define a width of "100% ". If the parent element does not specify a width then the rendering result may not represent what you expect.</p>	Sometimes obligatory	100 120 140 160 180 200 50% 100%
comment	Comment without any effect on rendering and behaviour. The comment is shown in the layout editor's tree view.	Optional	
Appearance			
width	(already explained above)		
size	Number of rows that are displayed inside the control. If specified as "1" (default) then the control is rendered as combo box - if ">1" then the control is rendered as multi line selection.	Optional	
displayonly	If set to true, the FIELD will not be accessible for input. It is just used as an output field.	Optional	
align	<p>Horizontal alignment of control in its column.</p> <p>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.</p> <p>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.</p>	Optional	left center right

valign	<p>Vertical alignment of control in its column.</p> <p>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. In this case the "align" property specifies the position of the control inside the column.</p>	Optional	<p>top</p> <p>middle</p> <p>bottom</p>
colspan	<p>Column spanning of control.</p> <p>If you use TR table rows then you may sometimes want to control the number of columns your control occupies. By default it is "1" - but you may want to define the control to span over more than one column.</p> <p>The property only makes sense in table rows that are synchronized within one container (i.e. TR, STR table rows). It does not make sense in ITR rows, because these rows are explicitly not synched.</p>	Optional	<p>1</p> <p>2</p> <p>3</p> <p>4</p> <p>5</p> <p>50</p> <p>int-value</p>
rowspan	<p>Row spanning of control.</p> <p>If you use TR table rows then you may sometimes want to control the number of rows your control occupies. By default it is "1" - but you may want to define the control to span over more than one row.</p> <p>The property only makes sense in table rows that are synchronized within one container (i.e. TR, STR table rows). It does not make sense in ITR rows, because these rows are explicitly not synched.</p>	Optional	<p>1</p> <p>2</p> <p>3</p> <p>4</p> <p>5</p> <p>50</p> <p>int-value</p>
renderasfield	<p>If set to "true" then the combo box is rendered like a FIELD control that offers valid value support.</p> <p>Default is "false".</p> <p>The normal translation of COMBODYN2 into HTML renders an HTML-select control. This control has certain limitations inside Internet Explorer: it only offers a very reduced set of styles to manipulate its look and feel and - much worse: it always occupies z-index "0" i.e. if you have other areas overlapping the COMBODYN2 area then COMBODYN2 is always on the top. This is quite ugly if e.g. a menu is opened and parts of the menu overlap a COMBODYN2 control.</p>	Optional	<p>true</p> <p>false</p>
allowmultiselection	<p>If set to true then multiple selections are allowed.</p>	Optional	<p>true</p> <p>false</p>

combostyle	<p>CSS style definition that is directly passed into this control.</p> <p>With the style you can individually influence the rendering of the control. You can specify any style sheet expressions. Examples are:</p> <p>border: 1px solid #FF0000</p> <p>background-color: #808080</p> <p>You can combine expressions by appending and separating them with a semicolon.</p> <p>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.</p>	Optional	
invisiblemode	<p>If the visibility of the control is determined dynamically by an adapter property then there are two rendering modes if the visibility is "false":</p> <p>(1) "invisible": the control is not visible.</p> <p>(2) "disabled": the control is deactivated: it is "grayed" and does not show any roll over effects any more.</p>	Optional	invisible cleared
tabindex	<p>Index that defines the tab order of the control. Controls are selected in increasing index order and in source order to resolve duplicates.</p>	Optional	-1 0 1 2 5 10 32767
<b>Binding</b>			
valueprop	(already explained above)		
validvaluesprop	(already explained above)		
displayprop	<p>Name of the adapter parameter that dynamically passes information whether the field is displayonly(true) or not (false).</p>	Optional	

statusprop	Name of the adapter parameter that dynamically passes information how the control should be rendered and how it should act. Use DISPLAYPROP to dynamically define whether the field is displayonly.	Optional	
titleprop	Name of the adapter parameter that dynamically defines the title of the control. The title is displayed as tool tip when the user moves the mouse onto the control.	Optional	
flush	<p>Flushing behaviour of the input control.</p> <p>By default an input into the control is registered within the browser client - and communicated to the server adapter object when a user e.g. presses a button. By using the FLUSH property you can change this behaviour.</p> <p>Setting FLUSH to "server" means that directly after changing the input a synchronization with the server adapter is triggered. As consequence you directly can react inside your adapter logic onto the change of the corresponding value. - Please be aware of that during the synchronization always all changed properties - also the ones that were changed before - are transferred to the adapter object, not only the one that triggered the synchronization.</p> <p>Setting FLUSH to "screen" means that the changed value is populated inside the page. You use this option if you have redundant usage of the same property inside one page and if you want to pass one changed value to all its representation directly after changing the value.</p>	Optional	screen server
flushmethod	When the data synchronization of the control is set to FLUSH="server" then you can specify an explicit event to be sent when the user updates the content of the control. By doing so you can distinguish on the server side from which control the flush of data was triggered.	Optional	
Online Help			
helpid	Help id that is passed to the online help management in case the user presses F1 on the control.	Optional	
titleprop	(already explained above)		
Miscellaneous			
testtoolid	Use this attribute to assign a fixed control identifier that can be later on used within your test tool in order to do the object identification	Optional	