MENU

This chapter covers the following topics:

- Example
- Separators
- Properties

Example

The example looks as follows:

When clicking on a menu item for which a function has been defined, then the name of the function is displayed in the status bar.

The XML layout definition is:

```
<page model="Menue_01_Adapter">
    <titlebar name="Menu Demo">
    </titlebar>
    <header align="left" withdistance="false">
        <menu menucollectionprop="menuData" width="100">
        </menu>
    </header>
    <pagebody>
    </pagebody>
    <statusbar withdistance="false">
    </statusbar>
</page>
```

In this example, the menu is embedded in the header. By the property `menucollectionprop`, it is bound to the adapter property `menuData`.

The Java code of the adapter is:

```
// This class is a generated one.

import com.softwareag.cis.server.Adapter;
import com.softwareag.cis.server.util.MENUNODEInfo;
import com.softwareag.cis.server.util.TREECollection;

public class Menue_01_Adapter
    extends Adapter
{
    // class >MenuDataItem<
    public class MenuDataItem extends MENUNODEInfo
    {
        public MenuDataItem(String text)
        {
            super(text);
        }
        public MenuDataItem(String text, String image)
        {
            super(text, image);
        }
        public void reactOnSelect()
        {
            outputMessage("S",getText() + " was called");
        }
    }

    // property >menuData<
    TREECollection m_menuData = new TREECollection();
    public TREECollection getMenuData() { return m_menuData; }

    /** initialisation - called when creating this instance*/
    public void init()
    {
        MenuDataItem top;
        top = new MenuDataItem("File");
        m_menuData.addTopNode(top,false);
        m_menuData.addSubNode(new MenuDataItem("New...","images/new.gif"),
        top,true,false);
```

```
        m_menuData.addSubNode(new MenuDataItem("Save","images/save.gif"),
        top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Save as..."),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("&SEPARATOR"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Remove"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("&SEPARATOR"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Exit"),top,true,false);
        top = new MenuDataItem("Edit");
        m_menuData.addTopNode(top,false);
        m_menuData.addSubNode(new MenuDataItem("Undo"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("&SEPARATOR"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Cut"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Copy"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("Paste"),top,true,false);
        top = new MenuDataItem("Help");
        m_menuData.addTopNode(top,false);
        m_menuData.addSubNode(new MenuDataItem("Online Help"),top,true,false);
        m_menuData.addSubNode(new MenuDataItem("About"),top,true,false);

    }
}
```

The member `m_menuData` holds an object of the instance `TREECollection`, which is defined in package `com.softwareag.cis.server.util`. The tree collection holds the menu items.

Each item is represented by an instance of the class `MyMENUNODEInfo`. The class itself is derived from the class `NODEInfo` in the package `com.softwareag.cis.server.util`. In the own class definition, the `reactOnSelect()` method is overwritten.

In the `init()` method of the class, the tree collection is assembled. Note that it can be reassembled at any point of time.

Separators

As you see in the example, separators can be added into the menu just as normal tree nodes, holding a special text "&SEPARATOR".

Properties

Basic			
menucollectionprop	Name of adapter property that represents the menu’s item hierarchy on server side. The property must be of type "TREECollection". Each menu item is represented by a tree node (subclassed from "NODEInfo") within the collection.	Obligatory	
comment	Comment without any effect on rendering and behaviour. The comment is shown in the layout editor’s tree view.	Optional	
Appearance			

width	Width of the control. There are three possibilities to define the width: (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. (B) Pixel sizing: just input a number value (e.g. "100"). (C) Percentage sizing: input a percantage 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.	Optional	100 120 140 160 180 200 50% 100%
height	Height of the control. There are three possibilities to define the height: (A) You do not define a height at all. As consequence the control will be rendered with its default height. If the control is a container control (containing) other controls then the height of the control will follow the height of its content. (B) Pixel sizing: just input a number value (e.g. "20"). (C) Percentage sizing: input a percantage value (e.g. "50%"). Pay attention: percentage sizing will only bring up correct results if the parent element of the control properly defines a height this control can reference. If you specify this control to have a height of 50% then the parent element (e.g. an ITR-row) may itself define a height of "100%". If the parent element does not specify a width then the rendering result may not represent what you expect.	Optional	
toggleimage	URL of the image that is shown on the right end of a menu item, if this item contains subitems. If not explicitly defined then a default icon is used.	Optional	
toggleimageprop	Name of adapter property that provides a URL-string that defines the toggle image. The toggle icon is shown on the right end of a menu item that has subitems.	Optional	

menustyle	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	
menustyleprop	Name of adapter property that dynamically provides explicit style information for the control.	Optional	