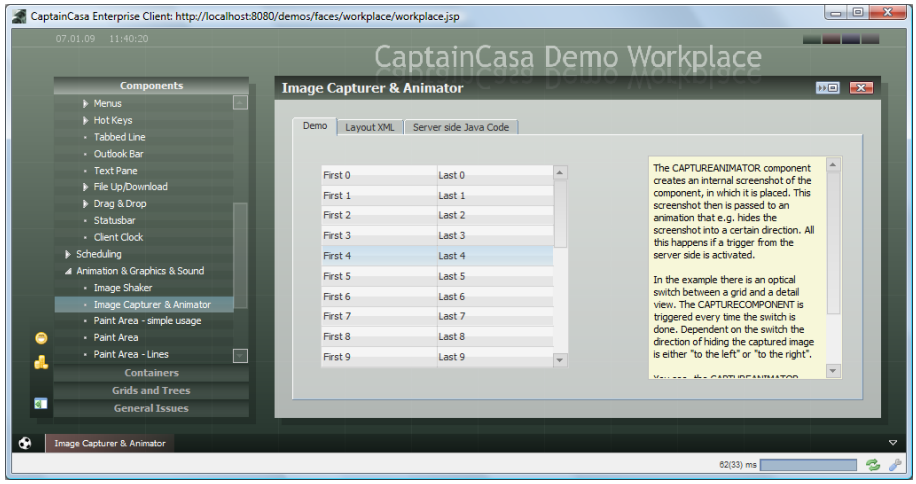
Component CAPTUREANIMATOR

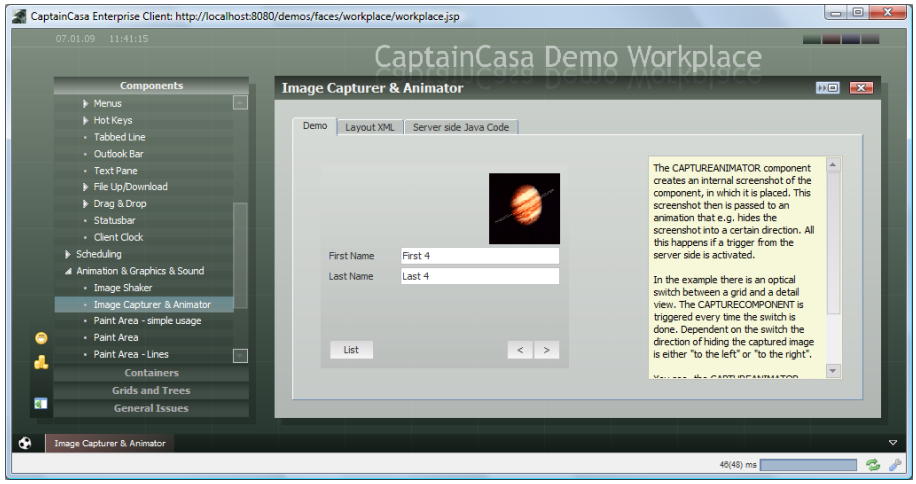
The component allows to animate a part of a screen in nice and simple way. It is used when the corresponding part's content changes. The CAPTUREANIMATOR component takes a screenshot of this part and slides the screenshot to the left, right, top or bottom.

Example Usage

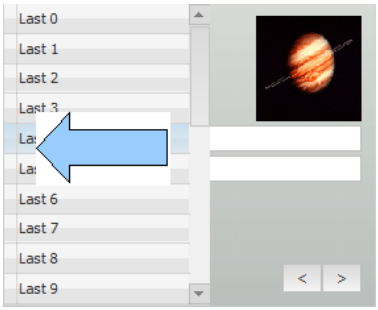
The following screen shots are taken from the demo workplace:



When the user clicks an item from the list then the corresponding pane's content is changed to a detail view:



The way the content changes is done in an animated way: the list is moved out of the pane to the left:



The same happens when the user uses the arrow-buttons within the detail view: the next

item is shown with the previous being moved to the left or the right, depending from the direction of navigation.

How to Use

The usage of the component is very simple – because the animation effect is only taking place within the client processing and because the animation is based on the screen shot mechanism that is used internally.

At the point of time when the content of the corresponding pane is changed the CAPTUREANIMATOR component receives a trigger: as result it takes a screenshot of the corresponding area and animates the screenshot according to an animation type definition:

```
<t:pane id="g_3" height="250" padding="10" rowdistance="5"
      width="300">

      <t:captureanimator id="g_4"
            animationtype="#{wp.DemoCaptureAnimator.animationType}"
            trigger="#{wp.DemoCaptureAnimator.trigger}" />

      <t:row id="g_5" rendered="#{wp.DemoCaptureAnimator.gridVisible}">
            <t:fixgrid id="g_6" background="#f4f4f4" border="#00000020"
                  bordercolor="#00000020" borderheight="1" borderwidth="1"
                  height="100%" ...>
                  ...
                  ...
                  ...   grid area definition
                  ...
                  ...
            </t:fixgrid>
      </t:row>

      <t:row id="g_11" rendered="#{wp.DemoCaptureAnimator.detailVisible}">
            <t:pane id="g_12"
                  height="100%" padding="10" rowdistance="3" width="100%">
                  <t:row id="g_13">
                  ...
                  ...
                  ... detail area definition
                  ...
                  ...
            </t:pane>
      </t:row>
</t:pane>
```

What you see in the layout definition: the outside pane is filled either with a row containing a gird area or with a row containing a detail pane. The visibility of both rows is controlled by a corresponding RENDERED attribute, so that only one of them shows up at a given point of time.

...and you see the CAPTUREANIMATOR definition, mainly consisting out of a binding to TRIGGER and a binding to an ANIMATIONTYPE.

The server side code that is executed when a grid selection is done looks like:

```
    private void select(MyRow r)
    {
        m_trigger.trigger();
        m_gridVisible = false;
        m_animationType = “hidetoleft”;
        m_selRow = r;
    }
```

The trigger is triggered, the animation type is set to “hidetoleft” and the flags internally managing the rendering of rows are set.

Without the CAPTUREANIMATOR component the outside pane would just change its content. Now, due to the CAPTUREANIMATOR it animates the previous content accordingly.

Further Information

Please check details within the demo workplace of update 20090107.