Special Issues

This chapter covers the following topics:

- Protocol Item
- Bringing Controls into the Layout Painter
- Text ID/Multi Language Controls

Protocol Item

Inside a tag handler, you receive a protocol item. There are some mandatory tasks that you have to do with a protocol item:

- You must tell the protocol item every property you are referencing from your control.

  This information is required because only these properties are transferred from the server to the client at runtime which are referenced inside the page.

- You must tell the protocol item every text ID you are referencing from your control.

  Again this information is used to send the right text IDs to the client processing.

In case of using macro controls, one macro control is rendered into many normal controls. Each normal control is treated in the way that it generates corresponding HTML/JavaScript and in the way that it itself tells which properties it binds to; i.e. each normal control adds its properties/text IDs itself: when your macro control contains some FIELD controls, then each FIELD control will tell during generation the adapter properties it binds to - there is no necessity for you to re-tell on macro control level.

But: you might tell on macro control level, that all the contained adapter properties are not provided via one-by-one implementation but by implementing a server side class already providing all sub-properties. In this case, you can use the protocol item in the following way:

- Call `addProperty('nameOfProperty','serverSideClass')`. For example:

  ```
  addProperty(m_addressprop,'ADDRESSInfo')
  ```

- Tell that all property definitions made by internally contained controls are not relevant for implementation by calling the method `suppressFurtherCodeGenEntries()`.

Bringing Controls into the Layout Painter

The Layout Painter is configured via a file *editor.xml* inside the *<installdir>/cis/config/* directory. This file contains information about all controls which are available inside the editor. For each control, the list of attributes and the list of possible subnodes is listed.

Have a look at the file - the structure is self-explaining.

With early versions, you had to bring own controls into the *editor.xml* file by editing it accordingly. The disadvantage was that every time Application Designer changed the *editor.xml* file, you had to reapply your changes. Application Designer now offers a dynamic way of adding own controls into the logical structure of the *editor.xml*.

Write an *editor_xyz.xml* file and place it into the same directory as *editor.xml*. "xyz" should be the same name as the one you chose as the prefix for your control library. Each *editor_xyz.xml* file holds information about the controls of the *xyz* control library:

- data types of a tag
- name of control tags
- attributes of tags
- subnodes a tag may have
- subnode extensions for existing Application Designer tags - this means, you define below which Application Designer controls your new tags should be positioned

The following definition shows the usage of the *editor_xyz.xml* file:

```
<!--
Dynamic extension of editor.xml file.
-->

<controllibrary>
        <editor>

                <!-- datatype TEXT -->
                <datatype name="demo:count">
                        <value id="1st" name="First"/>
                        <value id="2nd" name="Second"/>
                        <value id="3rd" name="Third"/>
                </datatype>

                <!-- control DEMOCONTROL -->
                <tag name="demo:democontrol">
                    <attribute name="text" datatype="demo:count"/>
                </tag>
                <tagsubnodeextension control="itr" newsubnode="demo:democontrol"/>
                <tagsubnodeextension control="tr" newsubnode="demo:democontrol"/>

                <!-- control DEMOCONTROLDYN -->
                <tag name="demo:democontroldyn">
                <attribute name="textprop"/>
                </tag>
                <tagsubnodeextension control="itr" newsubnode="demo:democontroldyn"/>
                <tagsubnodeextension control="tr" newsubnode="demo:democontroldyn"/>

                <!-- control ADDRESSROWAREA -->
                <tag name="demo:addressrowsarea">
                    <attribute name="addressprop"/>
                </tag>
                <tagsubnodeextension control="pagebody" newsubnode="demo:addressrowarea"/>

        </editor>
</controllibrary>
```

Note that the structure of the file directly corresponds to the structure of the original *editor.xml* file. The data is an add-on that is logically added to the information from the *editor.xml* file.

Note also that both new data types and new control tags are named together with their prefix - in order not to mix up with standard Application Designer controls or with controls of other control library providers.

Text ID/Multi Language Controls

Please contact Software AG in case you create new controls with language-dependent information - and if you want to use the same translation methods as Application Designer does for these controls.