Repeater

Repeater is a rich control with basic functionality of repeating a set of controls multiple times (just like for loop in traditional JSPs). The repeater is one-direction control; that is it only transfers data from the controller to the view (rendering). The repeater doesn't not bind post back data to the supplied list as JspxBean is doing.

The developer can define a variable to be used within the loop to bind child control attributes to the objects in the repeater list. Following is an example of the repeater:

```
<repeater id="customerRepeater" itemsList="customerList" var="c">
        <div>
                <p><label>${c.id}</label></p>
                <p><label>${c.name}</label></p>
        </div>
</repeater>
```

If the customerList in the controller contains 2 customers, the following will be rendered:

```
<div>
        <p><label>1</label></p>
        <p><label>customer 1</label></p>
</div>
<div>
        <p><label>2</label></p>
        <p><label>customer 2</label></p>
</div>
```

As we can see it repeated all children two times and evaluated each expression using the customer object in the corresponding loop cycle.

Repeater uses the same expression evaluation engine as JspxBean that expect the following format:

```
${variableName.propertyName}
```

Complex binding and super-class binding is also possible in the repeater context (see JspxBean complex binding and super-class binding for more info). The repeater can contain any control as a child control including another repeater control, which gives the ability to simulate nested loops as the following example.

```
<repeater id="customerRepeater" itemsList="customerList" var="c">
        <p><label>${c.name}</label></p>
        <repeater id="rep2" itemsList="c.creditCardList" var="user">
                <p><label>${user.cardName}</label></p>
        </repeater>
</repeater>
```

Attributes

Name	Description
itemList	The name of the list the repeater will loop against. The framework will try to get the list from the controller or from the parent repeater (see list binding below)
var	The variable name to be used in the binding of children
changeId	True or false (default is true), Boolean indicate if the framework should add the loop counter to the id of child controls to distinguish them from each other.

List Binding

As mentioned above the itemList attribute define the list name. The framework uses this name to get the list from the controller by invoking the public getter method of the list name (i.e. if the itemList=customerList, the framework will invoke getCustomerList in the controller).

In nested repeaters (example shown above), the list could also be obtained from binding expression pointing to the parent repeater variable. As you can see from the example above the second repeater list is bounded to the creditCardList of the customer, which is the loop variable of the parent repeater.

Change ids

In jspx, the control identifier is the id, so for any control that needs to be loaded in the post back, it must have a unique ID. The repeater control provides a feature of making sure of the uniqueness of the child control ids if required. This is done by changing the IDs of child controls during the rendering phase by adding the loop counter to the end of the original control. This behavior is controllable by the changeId attribute, if the changeId attribute is set to true (default) the repeater control will change the ids of all child controls unless the child control override this by setting its local changeId attribute to false. If the changeId attribute of the repeater is set to false, the repeater control will be not change the IDs of its child controls.

```
<repeater id="customerRepeater" itemsList="customerList" var="c">
        <p><label id="A">${c.id}</label></p>
        <p><label id="B" changeId="false">${c.name}</label></p>
</repeater>
```

Will be rendered as:

```
<p><label id="A_0">1</label></p>
<p><label id="B">customer 1</label></p>
<p><label id="A_1">2</label></p>
<p><label id="B">customer 2</label></p>
```

The above example shows a repeater with changeId attribute set to true (because the default is true), the first label changeId is true (default) while the second label override the changeId

attribute by setting it to false. The result shows that the first label ID was changed by adding the loop counter (A_0 and A_1) while the second label didn't (B).