generator

Description

NOTE: JSP-TAG

Generate an iterator based on the val attribute supplied.

NOTE: The generated iterator will **ALWAYS** be pushed into the top of the stack, and poped at the end of the tag.

Parameters

Dynamic Attributes Allowed:
false

Name	Required	Default	Evaluated	Type	Description
converter	false		false	org.apache.struts2.util.IteratorGenerator.Converter	The converter to convert the String entry parsed from *val* into an object
count	false		false	Integer	The max number entries to be in the iterator
separator	**true**		false	String	The separator to be used in separating the *val* into entries of the iterator
val	**true**		false	String	The source to be parsed into an iterator
var	false		false	String	The name to store the resultant iterator into page context, if such name is supplied

Examples

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

Example One:
<pre>
Generate a simple iterator
<s:generator val="%{'aaa,bbb,ccc,ddd,eee'}">
 <s:iterator>
     <s:property />
 </s:iterator>
</s:generator>
</pre>
This generates an iterator and print it out using the iterator tag.

Example Two:
<pre>
Generate an iterator with count attribute
<s:generator val="%{'aaa,bbb,ccc,ddd,eee'}" count="3">
 <s:iterator>
     <s:property />
 </s:iterator>
</s:generator>
</pre>
This generates an iterator, but only 3 entries will be available in the iterator
generated, namely aaa, bbb and ccc respectively because count attribute is set to 3

Example Three:
<pre>
Generate an iterator with var attribute
<s:generator val="%{'aaa,bbb,ccc,ddd,eee'}" count="4" separator="," var="myAtt" />
<%
 Iterator i = (Iterator) pageContext.getAttribute("myAtt");
 while(i.hasNext()) {
     String s = (String) i.next(); %>
     <%=s%>
<%   }
%>
</pre>
This generates an iterator and put it in the PageContext under the key as specified
by the var attribute.

Example Four:
<pre>
Generate an iterator with comparator attribute
<s:generator val="%{'aaa,bbb,ccc,ddd,eee'}" converter="%{myConverter}">
 <s:iterator>
     <s:property />
 </s:iterator>
</s:generator>

public class GeneratorTagAction extends ActionSupport {

  ....

  public Converter getMyConverter() {
     return new Converter() {
         public Object convert(String value) throws Exception {
             return "converter-"+value;
         }
     };
  }

  ...

}
</pre>
This will generate an iterator with each entries decided by the converter supplied. With
this converter, it simply add "converter-" to each entries.