

Iterator tool

On this page

- `$iterator.wrap(list)`
- `$<IteratorTool instance>.hasMore()`
- `$<IteratorTool instance>.more()`
- `$<IteratorTool instance>.remove()`
- `$<IteratorTool instance>.reset()`
- `$<IteratorTool instance>.stop()`
- `$<IteratorTool instance>.toString()`

`$iterator` is used with the **#foreach** loops. It wraps a list to let you specify a condition to terminate the loop and reuse the same list in a different loop. The following example shows how to use **`$iterator`**.

```
#set ($list = [1, 2, 3, 5, 8, 13])
#set ($numbers = $iterator.wrap($list))
#foreach ($item in $numbers)
  #if ($item < 8) $numbers.more()#end
#end
$numbers.more()
Output
-----
1 2 3 5
8
```

`$iterator.wrap(list)`

Wrap a list with the tool.

	Name	Type	Description
Parameter(s)	list	java.util.List	An array or a list instance.
Return	-	IteratorTool	Return an IteratorTool instance.

`$<IteratorTool instance>.hasMore()`

Check if the iteration has more elements.

	Name	Type	Description
Return	-	boolean	Return true if there are more elements in the wrapped list.

`$<IteratorTool instance>.more()`

Ask for the next element in the list.

	Name	Type	Description
Return	-	java.lang.Object	An element instance.

`$<IteratorTool instance>.remove()`

Remove the current element from the list.

`$<IteratorTool instance>.reset()`

Reset the wrapper so that it will start over at the beginning of the list.

`$<IteratorTool instance>.stop()`

Put a condition to break off a loop.

`$<IteratorTool instance>.toString()`

Return an object as a string.