Sun™ ONE Application Framework Tag Library Reference

Sun™ ONE Studio 5 update 1

Sun Microsystems, Inc.
www.sun.com

Part No. 817-4361-10
October 2003, Revision A

Submit comments about this document at: http://www.sun.com/hwdocs/feedback

---

Contents

Preface

This Sun™ ONE Application Framework Tag Library Reference is a brief introduction to the tag library, as well as a comprehensive reference to the tags available within the library.

How This Book Is Organized

Chapter 1, Overview

Chapter 2, containerView

Chapter 3, tiledView

Chapter 4, treeView

Chapter 5, useViewBean

Chapter 6, getDisplayFieldValue

Chapter 7, getModelFieldValue

Chapter 8, setDisplayFieldValue

Chapter 9, setModelFieldValue

Chapter 10, button

Chapter 11, checkbox

Chapter 12, combobox

Chapter 13, fileUpload

Chapter 14, form

Using UNIX Commands

This document might not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices. See the following for this information:

- Software documentation that you received with your system
- Solaris™ operating environment documentation, which is at

  `http://docs.sun.com`

Related Documentation

Application	Title	Part Number
Sun ONE Application Framework 2.1	*Sun ONE Application Framework Overview, Sun ONE Studio 5 update 1*	817-4360-10
Sun ONE Application Framework 2.1	*Sun ONE Application Framework Tutorial, Sun™ ONE Studio 5 update 1*	817-4358-10
Sun ONE Application Framework 2.1	*Sun ONE Application Framework IDE Guide, Sun ONE Studio 5 update 1*	817-4104-10
Sun ONE Application Framework 2.1	*Sun ONE Application Framework Developer's Guide, Sun ONE Studio 5 update 1*	817-4359-10
Sun ONE Application Framework 2.1	*Sun ONE Application Framework Component Author's Guide, Sun ONE Studio 5 update 1*	817-4362-10
Sun ONE Application Framework 2.1	*Sun ONE Application Framework Component Reference Guide, Sun ONE Studio 5 update 1*	817-4661-10

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

`http://www.sun.com/documentation`

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

http://www.sun.com/service/contacting

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

http://www.sun.com/hwdocs/feedback

Include the title and part number of your document with your feedback:

Sun ONE Application Framework Tag Library Reference, part number 817-4361-10

CHAPTER 1

Overview

Tag Library Overview

The JATO tag library is a powerful tool for rendering dynamic JSP output when used in conjunction with the JATO framework. This document is intended to be a brief introduction to the tag library, as well as a comprehensive reference to the tags available within the library. See supplemental JATO documentation for more complete information on writing a JATO application.

How the Tag Library Interacts With JATO

JATO is architected around the idea of arbitrarily nested `View` objects. Some of these views are `ContainerViews`, containing other `Views` (including other `ContainerViews`), and others are `DisplayFields`, with a notion of a value which can be accessed.

Each JSP has a single `ViewBean` associated with it. This object is also referred to as the *root view* of the JSP, as it is the top-level view object containing all other views used in that JSP. Each tag in the JATO tag library is a reference to one of these contained `Views`. The objects the tags refer to are called *peer views*, or simply *peers*. Each tag a developer uses in his or her JSP must refer to an existing peer object in that JSP's `ViewBean` or one of the `ViewBean`'s child views.

The bulk of the tags in the JATO tag library establish either a context within which other peers can be easily referenced, or they actually reference a peer view to render a visualHTML fragment such as a string value, a text field, or a list box. When a JSP using the tag library is rendered, it contains visual HTML form controls or other content derived from the interactions with each peer from within each tag. Thus, tags work in conjunction with a a set of predefined peers to render a dynamic HTML page from a JSP.

Using the Tag Library in Your Application

To use the JATO tag library within your Web application, each JSP using the tag library must reference the JATO Tag Library Descriptor, or TLD. This is accomplished by including the following directive at the top of your JSP:

```
<%@taglib uri="/WEB-INF/jato.tld" prefix="jato"%>
```

(Note: If you are using the Sun ONE Application Framework IDE toolset, this is automatically managed for you.) The location of the TLD specified by the uri attribute, as well as the tag prefix name, are arbitrary. However, it is recommended that you use the above values and including the JATO TLD file in the root of your application WAR file's `WEB-INF` directory. Within this document, assume the jato tag prefix.

Using the Tag Library in non-HTML JSPs

The JATO tag library contains both visual and non-visual tags. The visual tags render HTML 4.01 compliant markup, but non-visual tags have no associated markup, or allow developers to specify the markup they wish to render. Non-visual tags fall into several categories, but generally they represent the structure of your JATO `View` hierarchy in an abstract way. Because this hierarchy remains the same across content types, you can develop non-HTML markup-based JSPs using JATO, with the only additional requirement being able to specify visual markup for the appropriate content type.

The existing `getDisplayFieldValue` tag provides a way to easily inline dynamic display field data without any associated markup, while still firing display events and working with the current `ContainerView` context. This allows developers to add dynamic values to static markup declared in a JSP. Use of this tag is an easy way to develop non-HTML (or even HTML) pages without creating new JSP tags. However, in general, creating new JSP tags is the easier approach to render non-HTML JSPs.

Beginning with JATO 2.0, component libraries gave developers the ability to easily package and deliver components that automatically manage their own tags for multiple content types, whether they are the standard JATO tag library tags or custom-built tags. Furthermore, beginning with JATO 2.1, JATO's `taglib` package has been reimplemented to provide much easier ways for developers to create new custom tags that render View components in arbitrary ways. Using these tools, it is easy for developers to either find an existing component library that handles visual markup of the content type they require, or create a component library that renders components using non-HTML content types. See the documentation for the taglib package for more information on creating your own visual tags.

Including JSP Content: Pagelets

JSPs allows inclusion of outside content in the currently rendering page (see the JSP 1.1 specification for full details). This allows developers to modularize JSP content and then combine this content into compound documents. This capability has interesting implications for JATO applications.

There are two ways to include content in a JSP: translation-time includes, and request-time includes. A translation-time include pulls an outside file's content into the enclosing JSP before it is translated into a servlet by the container. The benefit of this approach is that it performs well, and the included content acts just like it were part of the enclosing page. The downside is that it is not at all dynamic—the included content is statically enclosed and cannot be chosen or replaced at runtime.

A request-time include does a `RequestDispatcher.include()` operation on a target URL/JSP, dynamically inlining the target content into the enclosing page. This include is done every time the enclosing page is rendered. The benefit of this approach is that the included file can be chosen dynamically. The downside is that there is some performance overhead in dispatching the request to the included resource.

It is generally advocated *translation-time includes* of JSP fragments in JATO applications because they perform well, and can include just the views/fields appropriate for the scope of the inclusion. This approach lets a component writer compose a fragment of JSP/JATO content and conveniently reuse it in several pages. But, as noted above, the choice of which fragment to include in the enclosing page is determined at translation time and is then fixed, significantly limiting the dynamism of the rendered page.

Request-time includes in JATO applications would be a nice way to get around the static nature of *translation-time includes*, except that in prior versions of JATO, there was not any way to request-time include anything but a full ViewBean and its contents. This meant that developers had to basically include a root view inside another root view, which was both confusing and not always guaranteed to work correctly—there should only be one root view per logical page. The other downside to that approach is that the included content lost the scope of the enclosing JSP's container view; the scope of objects accessed in the included page is limited by the immediately enclosing ViewBean, meaning that the page cannot necessarily be arbitrarily included in other JSPs like a translation-time include would allow.

Since JATO 1.2, a solution has been offered to these limitations in the form of *pagelets*, accompanied by a `<jato:pagelet>` tag. In JATO, a pagelet is distinguished from an arbitrary non-pagelet JSP fragment by the fact that it can be seamlessly request-time included in a JATO JSP. This is accomplished by the use of the `<jato:pagelet>` tag, which *connects* the view tags in the included JSP to the enclosing JSP's container scope, so that an included pagelet uses the enclosing JSP's current container view as its container view scope.

For example, assume you have the following JSP and pagelet:

EnclosingPage.jsp

```
<%@page info="E0130" language="java"%>
<%@taglib uri="/WEB-INF/jato.tld" prefix="jato"%>

<jato:useViewBean ...>

<jato:containerView name="foo">
    ...
    <jsp:include page="MyPagelet.jsp"/>
    ...
</jato:containerView>

</jato:useViewBean>
```

MyPagelet.jsp

```
<%@page info="MyPagelet" language="java"%>
<%@taglib uri="/WEB-INF/jato.tld" prefix="jato"%>

<jato:pagelet>
    <jato:combobox name="month"/>...
</jato:pagelet>
```

The “month” field sees container view “foo” as its enclosing container, and thus can be declared a proper child of “foo”. The `<jato:pagelet>` tag acts as a proxy to the enclosing page's container view tag, connecting the two JSPs as though they were part of the same rendering, when in fact they are two separate renderings.

Even more interesting, developers can use the display events in JATO 1.2 to dynamically choose which pagelet to include at request time (by using the `PageContext.include()` method in the event). This technique is demonstrated in the JATO sample application.

The following matrix summarizes the types of inclusion techniques developers can use in JATO:

JSP Fragment Type	Translation-time Include	Runtime Include
Pagelet	`Yes*`	`Yes`
Fragment (no ViewBean)	`Yes`	`No`
Fragment (with ViewBean)	`No`	`Yes`

* Technically, this works; however, there is no reason other than consistency to use the `<jato:pagelet>` tag in this situation, since the included JSP will simply become part of the enclosing JSP. The pagelet tag is written to do nothing in this situation.

Tag Overview

The tags in the JATO tag library fall into three basic groups: context tags, value tags, and visual tags. See the following sections for more information.

Context Tags

These tags are oriented toward declaring a peer view's scope, within which the referenced object will define the current context for other embedded tags. Specifically, this means that each of these tags declare usage of a `ContainerView` (or a specialization of `ContainerView`). Each such declaration establishes a name scope in which child Views can be referred to by their short, non-qualified names. `ContainerView` contexts may be nested, and do not directly result in any rendered HTML.

Tag Name	Alternate Names	Description
containerView		Declare usage of `ContainerView`
tiledView		Declare usage of a `TiledView`
treeView		Declare usage of a `TreeView`
useViewBean	viewBean	Declare usage of a `ViewBean`

Value Tags

These tags allow for direct manipulation of DisplayField and Model values. Developers may embed these tags in scriptlets, expressions, or HTML within the JSP. Unlike the visual tags described below, these tags do not render an HTML form control; rather, these tags manipulate values directly. The value tags can be used to get/set values on any of the DisplayField subtypes, or any Model instance available in the application. These tags must appear with the scope of a context tag.

Tag Name	Description
getDisplayFieldValue	Retrieve a value from a DisplayField
getModelFieldValue	Retrieve a value from a Model field
setDisplayFieldValue	Set the value of a DisplayField
setModelFieldValue	Set the value of a Model field

Visual (HTML) Tags

These tags use a a combination of tag attributes and values from DisplayField peers to render HTML form controls. These tags must also appear with the scope of a context tag.

Tag Name	Alternate Names	Description
button		Render a button control
checkbox	checkBox	Render a checkbox control
combobox	comboBox	Render a combobox control
fileUpload		Render a file upload element
form		Define an HTML form
frameSrc		Render a frame source element
hidden		Render a hidden element
href		Render a hyperlink element
image		Render an image
listbox	listBox	Render a list control
password		Render a password control
radioButtons		Render a group of radio button controls
text	staticText	Render arbitrary text

Tag Name	Alternate Names	Description
textArea		Render a multi-line text area control
textField		Render a single line text control
validatingTextArea		Render a multi-line text area control
validatingTextField		Render a single line text control

Tree Tags

These tags can be combined with the **treeTag** to specify tree rendering logic.

Tag Name	Description
treeLevel	Denote a content section that will be rendered “level-times” for matching tree levels
treeNode	Denote a content section that will be rendered for matching tree nodes
treeNodeHandle	Renders a expand/collapse node control

Miscellaneous Tags

These tags provide additional features.

Tag Name	Description
content	Denote a content section and associate it with a display event
pagelet	Allows JATO tags in an included JSP fragment to inherit the enclosing page’s container view scope, and thus be included at request-time

Tag Reference

Note Legend:

RTExpr = Attribute value can be a runtime-evaluated expression

Req = Required attribute

CHAPTER 2

containerView

<jato:containerView>

A containerView declares use of a `ContainerView` peer instance available from the view bean or one of its child `ContainerViews`. All `View` or `DisplayField` references enclosed by this tag are assumed to reference children of the associated `ContainerView` instance, and their `name` attributes are resolved within the namespace of the parent container. This tag is only valid when nested inside a useViewBean, tiledView, or another containerView tag.

The use of the containerView tag causes a new scripting variable to be defined within the scope of the enclosed tag body. The name of this scripting variable is the value of the `name` attribute, or the name of the `id` attribute if specified. When processing a `containerView` tag, the implicit scripting variable `currentContainerView` is set to the `ContainerView` instance referred to by this tag.

The useViewBean and tiledView tags are special cases of the containerView tag, adding additional behavior to the basic behavior defined by this tag.

Example:

```
<jato:containerView name="header">

...

</jato:containerView>
```

Attribute Name	Description	Notes
id	Specifies a name used to identify the implicit scripting variable that will refer to the `ContainerView` instance. The name specified is case sensitive and shall conform to the current scripting language variable-naming conventions. If not specified, the scripting variable is named with the value specified in the "name" attribute.	
name	The name of the `ContainerView` peer instance as declared in its parent (a `ContainerView`, `TiledView`, or `ViewBean`). This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req, RTExpr
type	If specified, defines the type of the scripting variable defined within the scope of this tag. This allows the type of the scripting variable to be distinct from, but related to, that of the implementation class specified. The type is required to be either the specific `ContainerView` class itself, a superclass of the class, or an interface implemented by the class specified. The object referenced is required to be of this type, otherwise a `java.lang.ClassCastException` shall occur at request time when the assignment of the object referenced to the scripting variable is attempted. If unspecified, the scripting variable is declared to be of type `<JATO package>.view.ContainerView`	

CHAPTER 3

tiledView

<jato:tiledView>

A tiledView tag declares a block of body content which is evaluated and rendered repeatedly during the display of the JSP. The tiledView must be associated with a `TiledView` object available from the view bean or one of its child `ContainerViews`. Callbacks to the `TiledView` object control the display of the tiledView body content. A tiledView is particularly useful for displaying multiple rows an HTML table; however, a tiledView's usage is not restricted to dynamic HTML row generation.

The use of the tiledView tag causes a new scripting variable to be defined within the scope of the enclosed tag body. The name of this scripting variable is the value of the `name` attribute, or the name of the `id` attribute if specified. When processing a tiledView tag, the implicit scripting variables `currentContainerView` and `currentTiledView` are set to the `TiledView` instance referred to by this tag.

A tiledView tag can be considered a special case of containerView tag. This tag is only valid when nested inside a useViewBean, containerView, or another tiledView tag.

Example:

```
<jato:tiledView name="orderItems" maxTiles="10">

...

</jato:tiledView>
```

Attribute Name	Description	Notes
id	Specifies a name used to identify the implicit scripting variable that will refer to the TiledView instance. The name specified is case sensitive and shall conform to the current scripting language variable-naming conventions. If not specified, the scripting variable is named with the value specified in the name attribute.	
maxTiles	Valid values: positive number Specifies the maximum number of times this TiledView will render its body content. This value will override the corresponding value in the TiledView object. If not specified, maxTiles is determined by the TiledView peer	RTExpr
name	The name of the TiledView peer instance as declared in its parent (a ContainerView, TiledView, or ViewBean). This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a ContainerView or a derivative of ContainerView (such as TiledView). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the ViewBean). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: /header/orderList/customerName (absolute from root view) orderList/customerName (relative to current container) ../footer/orderList/customerName (relative to parent)	Req, RTExpr
type	If specified, defines the type of the scripting variable defined within the scope of this tag. This allows the type of the scripting variable to be distinct from, but related to, that of the implementation class specified. The type is required to be either the specific TiledView class itself, a superclass of the class, or an interface implemented by the class specified. The object referenced is required to be of this type, otherwise a java.lang.ClassCastException shall occur at request time when the assignment of the object referenced to the scripting variable is attempted. If unspecified, the scripting variable is declared to be of type <JATO package>.view.TiledView	
reset	Valid values: “true” or “false” A value of “true” specifies that the tag will call the TiledView's resetTileIndex() method immediately after invoking the beginDisplay() event. This will ensure that the view's primary model is properly reset for display. If not specified, a value of “true” is assumed. Note, this is a change from previous versions of JATO. Previous versions never performed a reset, leaving this behavior up to the developer. It is expected that this change will provide more ease of use, while providing backward compatibility for the rare application that relied upon the previous default behavior.	

CHAPTER 4

treeView

<jato:treeView>

A treeView tag declares a block of body content which is evaluated and rendered repeatedly during the display of the JSP. The treeView must be associated with a `TreeView` object available from the view bean or one of its child `ContainerViews`. Callbacks to the `TreeView` object control the display of the treeView body content. A treeView is particularly useful for displaying hierarchical data. Typically, though not by definition, a **treeView** tag will contain one or more treeNode tags.

The use of the treeView tag causes a new scripting variable to be defined within the scope of the enclosed tag body. The name of this scripting variable is the value of the `name` attribute, or the name of the `id` attribute if specified. When processing a treeView tag, the implicit scripting variables `currentContainerView` and `currentTreeView` are set to the `TreeView` instance referred to by this tag.

A treeView tag can be considered a special case of containerView tag. This tag is only valid when nested inside a useViewBean, containerView, tiledView, or another treeView tag.

See also treeNode, treeLevel, treeNodeHandle

Example:

```
<jato:treeView name="treeMenu">

...

</jato:treeView>
```

Attribute Name	Description	Notes
id	Specifies a name used to identify the implicit scripting variable that will refer to the `ContainerView` instance. The name specified is case sensitive and shall conform to the current scripting language variable-naming conventions. If not specified, the scripting variable is named with the value specified in the “name” attribute.	
name	The name of the `TreeView` peer instance as declared in its parent container view. This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req, RTExpr
type	If specified, defines the type of the scripting variable defined within the scope of this tag. This allows the type of the scripting variable to be distinct from, but related to, that of the implementation class specified. The type is required to be either the specific `ContainerView` class itself, a superclass of the class, or an interface implemented by the class specified. The object referenced is required to be of this type, otherwise a `java.lang.ClassCastException` shall occur at request time when the assignment of the object referenced to the scripting variable is attempted. If unspecified, the scripting variable is declared to be of type `<JATO package>.view.TreeView`.	
reset	Valid values: “true” or “false” A value of “true” specifies that the tag will call the `TreeView`'s `resetNodeLocation()` method immediately after invoking the `beginDisplay()` event. This will ensure that the view's primary model is properly reset for display. If not specified, a value of “true” is assumed. Note, this is a change from previous versions of JATO. Previous versions never performed a reset, leaving this behavior up to the developer. It is expected that this change will provide more ease of use, while providing backward compatibility for the rare application that relied upon the previous default behavior.	

CHAPTER 5

useViewBean

<jato:useViewBean>

A useViewBean tag establishes the root ViewBean peer used for rendering the current JSP. Only one useViewBean tag is permitted within a JSP, and it must enclose all other JATO tags on that page. The useViewBean tag establishes a root name space for a given JSP and acts as controller during rendering of the page.

The use of the useViewBean tag causes a new scripting variable called viewBean to be defined within the scope of the enclosed tag body. When processing a useViewBean tag, the implicit scripting variables currentContainerView and currentTiledView are established. The value of the currentContainerView variable is set to the ViewBean instance referred to by this tag.

A useViewBean tag can be considered a special case of containerView tag.

Example:

```
<jato:useViewBean className="com.mycomp.myapp.MyViewBean">

...

</jato:useViewBean>
```

Attribute Name	Description	Notes
className	The fully qualified name of the `ViewBean` peer class. The class name is case sensitive.	Req
type	If specified, defines the type of the `viewBean` scripting variable defined within the scope of this tag. This allows the type of the scripting variable to be distinct from, but related to, that of the implementation class specified. The type is required to be either the specific ViewBean class itself, a superclass of the class, or an interface implemented by the class specified. The object referenced is required to be of this type, otherwise a `java.lang.ClassCastException` shall occur at request time when the assignment of the object referenced to the scripting variable is attempted. If unspecified, the scripting variable is declared to be of the type specified by the `className` attribute.	

CHAPTER 6

getDisplayFieldValue

<jato:getDisplayFieldValue>

This tag invokes the `DisplayField.getValue()` method on the named display field and inlines the resulting value in the HTML output stream. This tag may be embedded within any arbitrary HTML to manually construct HTML form controls or provide values to other tag attributes that allow runtime value expressions.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:getDisplayFieldValue name="salutation" defaultValue=
"Mr."/>
```

Attribute Name	Description	Notes
defaultValue	The value to display if the `DisplayField` peer value is null	RTExpr
escape	Valid values: “true” or “false” A value of “true” specifies that characters with special meaning in HTML (such as < and >) will be changed to the equivalent entity representation before being output. If not specified, a value of “true” is assumed.	
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req, RTExpr

CHAPTER 7

getModelFieldValue

<jato:getModelFieldValue>

This tag invokes the `Model.getValue()` method on the named model instance and inlines the resulting value in the HTML output stream. This tag may be embedded within any arbitrary HTML to manually construct HTML form controls or provide values to other tag attributes that allow runtime value expressions.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:getModelFieldValue modelClass=
"com.mycompany.myapp.MyModel" name="salutation" defaultValue=
"Mr."/>
```

Attribute Name	Description	Notes
defaultValue	The value to display if the Model field value is null	RTExpr
escape	Valid values: “true” or “false” A value of “true” specifies that characters with special meaning in HTML (such as < and >) will be changed to the equivalent entity representation before being output. If not specified, a value of “true” is assumed.	
lookInSessio n	Valid values: “true” or “false” A value of “true” specifies that the tag will specify to the ModelManager that it should look in the session for the model named by the modelName attribute. The modelName attribute must be specified if this attribute is specified. If not specified, “false” is assumed.	RTExpr
modelClass	The fully-qualified model class interface or implementation name (see the documentation for <JATO package>.ModelManager for more information on which class to specify). The instance returned from the ModelManager in response to this class name must implement the <JATO package>.model.Model interface.	Req, RTExpr
modelName	If specified, attempts to obtain the model named by the value of this attribute. If this attribute is not specified, a default model instance will be used. This attribute is mandatory if the lookInSession attribute is set to “true”. If the lookInSession attribute is “true”, the ModelManager will attempt to obtain the model from the session using the name specified by this attribute.	RTExpr
name	The name of the model field.	Req, RTExpr

CHAPTER 8

setDisplayFieldValue

<jato:setDisplayFieldValue>

This tag invokes the `DisplayField.setValue(Object)` method with the provided value. This tag does not render any HTML, and may be embedded within any arbitrary HTML on the page.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:setDisplayFieldValue name="numItems" value="8"
valueType="int"/>
```

Attribute Name	Description	Notes
name	The name of the DisplayField peer. This peer must be a child of the current parent ContainerView, TiledView, or ViewBean. The DisplayField name is resolved relative to the current parent view. The view peer must be of type <JATO package>.view.DisplayField This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a ContainerView or a derivative of ContainerView (such as TiledView). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the ViewBean). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: /header/orderList/customerName (absolute from root view) orderList/customerName (relative to current container) ../footer/orderList/customerName (relative to parent)	Req, RTExpr
value	The value that will be passed into the DisplayField.setValue(Object) method of the specified DisplayField peer.	Req RTExpr
valueType	Valid values: “string”, “int”, “long”, “float”, “double”, “short”, “bigdecimal”, “boolean”, “byte”, “char”, “sqldate”, “sqltime”, “sqltimestamp” The type to which the string representation of the value attribute will be converted before being set on the target DisplayField. The type conversion from the string representation must be legal or an exception will be thrown. If this attribute is omitted, the set value will be of type String.	

CHAPTER 9

setModelFieldValue

<jato:setModelFieldValue>

This tag invokes the `Model.setValue(String,Object)` method with the provided value. This tag does not render any HTML, and may be embedded within any arbitrary HTML on the page.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:setModelFieldValue modelClass=
"com.mycompany.myapp.MyModel" name="numItems" value="8"
valueType="int"/>
```

Attribute Name	Description	Notes
lookInSession	Valid values: “true” or “false” A value of “true” specifies that the tag will specify to the `ModelManager` that it should look in the session for the model named by the `modelName` attribute. The `modelName` attribute must be specified if this attribute is specified. If not specified, “false” is assumed.	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField`	Req, RTExpr
value	The value that will be passed into the `DisplayField.setValue(Object)` method of the specified `DisplayField` peer.	Req RTExpr

Attribute Name	Description	Notes
modelClass	The fully-qualified model class interface or implementation name (see the documentation for `<JATO package>.ModelManager` for more information on which class to specify). The instance returned from the `ModelManager` in response to this class name must implement the `<JATO package>.model.Model` interface.	Req, RTExpr
modelName	If specified, attempts to obtain the model named by the value of this attribute. If this attribute is not specified, a default model instance will be used. This attribute is mandatory if the `lookInSession` attribute is set to "true". If the `lookInSession` attribute is "true", the `ModelManager` will attempt to obtain the model from the session using the name specified by this attribute.	RTExpr
valueType	Valid values: "string", "int", "long", "float", "double", "short", "bigdecimal", "Boolean", "byte", "char", "sqldate", "sqltime", "sqltimestamp" The type to which the string representation of the value attribute will be converted before being set on the target `DisplayField`. The type conversion from the string representation must be legal or an exception will be thrown. If this attribute is omitted, the set value will be of type `String`.	

CHAPTER 10

button

<jato:button>

Renders an HTML `<input type="submit">` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:button name="processForm"/>
```

The above would be rendered into the following HTML:

```
<input type="submit" name="PageFoo.processForm" value=
"Process">
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a button tag, “value” refers to the value that will be rendered in the HTML element's “value” attribute	RTExpr
disabled	Valid values: “true” or “false” Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, “false” is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The DisplayField name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.CommandField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
src	An image URL if this button is to be rendered as an image button.	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr

Attribute Name	Description	Notes
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp	

CHAPTER 11

checkbox

<jato:checkbox>

Renders an single HTML `<input type="checkbox">` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:checkbox name="sendAdditionalInfo"/>
```

The above would be rendered into the following HTML:

```
<input type="checkbox" name="PageFoo.sendAdditionalInfo"
value="true">
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
label	The label for the checkbox which will be rendered to the right of the control	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.BooleanDisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
style	CSS styles to be applied to this HTML element.	RTExpr

Attribute Name	Description	Notes
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 12

combobox

<jato:combobox>

Renders an HTML combobox as comprised of an HTML <input type="select"> element containing an arbitrary number of HTML <option> elements. This tag offers the convenience of treating the combobox as a single element and the flexibility of dynamic list content generation.

This tag is only valid when enclosed by both an HTML <form> element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:combobox name="itemCondition"/>
```

The above would be rendered into the following HTML:

```
<select name="PageFoo.itemCondition">
  <option value="" selected>None Selected</option>
  <option value="Excellent">Excellent</option>
  <option value="Good">Good</option>
  <option value="Average">Average</option>
  <option value="Poor">Poor</option>
</select>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.html.SelectableGroup` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
style	CSS styles to be applied to this HTML element.	RTExpr

Attribute Name	Description	Notes
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	RTExpr
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 13

fileUpload

<jato:fileUpload>

Renders an HTML `<input type="file">` element. Used in conjunction with the `com.iplanet.jato.MultipartFormServletFilter` class, allows developers to easily provide file upload capabilities within their application.

The fileUpload tag must be used within a form declared with the `multipart/form-data` content type:

```
<jato:form name="form1" method="post" encType="multipart/form-
data">

<jato:fileUpload name="fileUpload1"/>

</jato:form>
```

This tag may not contain body content.

Example:

```
<jato:fileUpload name="fileUpload1" target="_top"/>
```

The above would be rendered into the following HTML:

```
<input type="file" name="Page1.fileUpload1">
```

Attribute Name	Description	Notes
accept	Specifies a comma-separated list of content types that a server processing this input tag will handle correctly. User agents may use this information to filter out non-conforming files when prompting a user to select files to be sent to the server.	
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
maxLength	Valid values: positive number Specifies the maximum number of characters the user may enter in the field. If not specified, an unlimited number of characters may be entered into the field.	RTExpr
name	The name of the `FileUpload` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.html2.FIleUpload` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
size	The size of the file input text field	RTExpr

Attribute Name	Description	Notes
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 14

form

<jato:form>

Renders an HTML `<form>` element. The form contents are described by the body content of this tag. The HTML form attribute action is not exposed in this tag because the tag handler dynamically constructs this attribute to be appropriate for the JATO runtime environment.

Note, the form tag does not have a view peer, nor can it be nested inside another JATO or HTML `<form>` element. However, it must be enclosed ultimately by a useViewBean tag. This tag may contain body content.

Example:

```
<jato:form name="form1" target="_top"/>
```

The above would be rendered into the following HTML:

```
<form name="form1" method="post" action="...(JATO URL)...">

...

</form>
```

Property Name	Description	Notes
accept	Specifies a comma-separated list of content types that a server processing this form will handle correctly. User agents may use this information to filter out non-conforming files when prompting a user to select files to be sent to the server.	
acceptCharset	Specifies the list of character encodings for input data that is accepted by the server processing this form. The value is a space- and/or comma-delimited list of charset values. The client must interpret this list as an exclusive-or list, i.e., the server is able to accept any single character encoding per entity received. The default value for this attribute is the reserved string "UNKNOWN". User agents may interpret this value as the character encoding that was used to transmit the document containing this FORM element.	
defaultCommandChild	The name of a `CommandField` child which will be activated during a default request. A default request is a request that does not include specification of an activated command child. Such a situation will typically occur when the user presses the "Enter" key while in a form field, thereby resulting in submission of the enclosing form without choosing a particular button. The view peer must be of type `<JATO package>.view.CommandField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `defaultCommandChild="/okButton"` (absolute from root view) `defaultCommandChild="okButton"` (relative to current container) `defaultCommandChild="../okButton"` (relative to parent)	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
encType	Specifies the content type used to submit the form to the server (when the value of method is "post").The default value for this attribute is "application/x-www-form-urlencoded". The value "multipart/form-data" should be used in combination with the INPUT element, type="file".	RTExpr

Property Name	Description	Notes
method	Valid values: "post" or "get" The HTTP method that will be used to submit this request. If not specified, value will be "post".	
name	The name of the form element within the enclosing HTML document. This name is arbitrary and not related to any peer instance	
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
target	Window target to which this form is submitted.	RTExpr
	This tag also supports the following JavaScript events: onReset, onSubmit	

CHAPTER 15

frameSrc

<jato:frameSrc>

Renders the src and name attributes for a HTML <frame> tag.

This tag is only valid when enclosed by both an HTML <form> element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

NOTE: This tag does not attempt to entirely replace the HTML <frame> tag because HTML editors will not allow you to edit your frameset in a WYSIWYG fashion without the presence of HTML <frame> tags. Therefore, this tag is designed to supply the HTML <frame> tag with a dynamic src value only.

Example:

```
<frameset rows="30%,70%">

  <frame <jato:frameSrc name="Frame1" location="internal" />
  >

  <frame <jato:frameSrc name="Frame2" location="external" />
  >

</frameset>
```

The above would be rendered into the following HTML:

```
<frameset rows="30%,70%">

  <frame src="...(JATO page)..." name="Frame1" >

  <frame src="/blank.html" name="Frame2" >

</frameset>
```

Attribute Name	Description	Notes
defaultValue	The value to use, if the DisplayField view peer's value is null. For a frameSrc tag, "value" refers to a JATO page name when the location attribute is "internal". If the location attribute is "external", then the "value" refers to an arbitrary non-JATO URL.	RTExpr
name	The name of the DisplayField peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
location	Valid values: "internal" or "external" A value of "internal" specifies that the source document for this frame will be a JATO page in the current application. A value of "external" specifies that the source document for this frame will be an arbitrary URL external to the current application. If not specified, "internal" is assumed.	
	This tag does not support any JavaScript events. You can add JavaScript directly in the HTML `<frame>` element that contains this tag.	

CHAPTER 16

hidden

<jato:hidden>

Renders an HTML `<input type="hidden">` element.

This tag is only valid when enclosed by both an HTML <form> element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:hidden name="nameFilter" defaultValue="*"/>
```

The above would be rendered into the following HTML:

```
<input type="hidden" name="PageFoo.nameFilter" value="Jo*">
```

Attribute Name	Description	Notes
defaultValue	The value to use, if the DisplayField view peer's value is null. For a hidden tag, "value" refers to the value that will be rendered in the HTML Hidden element's "value" attribute	RTExpr
name	The name of the DisplayField peer. This peer must be a child of the current parent ContainerView, TiledView, or ViewBean. The DisplayField name is resolved relative to the current parent view. The view peer may be any subtype of <JATO package>.view.DisplayField This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a ContainerView or a derivative of ContainerView (such as TiledView). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the ViewBean). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: /header/orderList/customerName (absolute from root view) orderList/customerName (relative to current container) ../footer/orderList/customerName (relative to parent)	Req
	This tag does not support any JavaScript events.	

CHAPTER 17

href

<jato:href>

Renders an HTML `<a href="..." >...</a>` element.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag. This tag may contain body content. This content will be included between the `<a>` and `</a>` tags of the rendered element and thus appear as the visible, clickable href element.

Example:

```
<jato:href name="orderDrillDown">See orders</jato:href>
```

The above would be rendered into the following HTML:

```
<a href="...(JATO URL)...?PageFoo.orderDrillDown=10345">See
orders</a>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
anchor	The anchor value to append to the generated HREF.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr

Attribute Name	Description	Notes
defaultValue	The value to use, if the `DisplayField` view peer's value is null. For an href tag, "value" refers to a special name-value pair that will be automatically included as part of the HREF's query string. The format for the implicit name-value pair is the following: `...?<qualified display field name>=<value>&...` This mechanism allows an href to have a display string as well as a value important to the application.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.CommandField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
queryParams	Specifies an arbitrary number of name value pairs which will be appended to the HREF's query string. The name value pairs should follow the format "name=value", where value is URL encoded by the JSP author as needed. Additional name value pairs must be delimited by the '&' character. Example: queryParams="fname=Mike&lname=Jones"	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr

Attribute Name	Description	Notes
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
target	A frame target	RTExpr
title	Offers advisory information about this element.	RTExpr
trim	Valid values: "true" or "false" A value of "true" specifies that the tag content should be trimmed of all leading and trailing whitespace. A value of "false" preserves the whitespace. If not specified, "true" is assumed.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 18

image

<jato:image>

Renders an HTML `<img>` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:image name="employeePhoto" defaultValue=
"/images/nophoto.jpg"/>
```

The above would be rendered into the following HTML:

```
<img name="PageFoo.employeePhoto" src="...(image URL)...">
```

Attribute Name	Description	Notes
align	Specifies the position of an IMG with respect to its context. The following values for align concern the object's position with respect to surrounding text: • bottom: means that the bottom of the object should be vertically aligned with the current baseline. This is the default value. • middle: means that the center of the object should be vertically aligned with the current baseline. • top: means that the top of the object should be vertically aligned with the top of the current text line. • left causes the image to float to the current left margin. • right causes the image to float to the current right margin.	RTExpr
alt	Specifies alternate text for user agents that cannot display the image	RTExpr
border	Specifies the width of the IMG border, in pixels. The default value for this attribute depends on the user agent.	RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For an image tag, "value" refers to the value that will be rendered in the HTML img element's `src` attribute	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
height	Image height override. When specified, the width and height attributes tell user agents to override the natural image or object size in favor of these values.	RTExpr
hspace	Specifies the amount of white space to be inserted to the left and right of the IMG. The default value is not specified, but is generally a small, non-zero length.	RTExpr
isMap	Valid values: "true" or "false" Specifies that the IMG is a server side image map. When the user activates the link by clicking on the image, the screen coordinates are sent directly to the server where the document resides. Screen coordinates are expressed as screen pixel values relative to the image. If not specified, "false" is assumed.	RTExpr
longDesc	Specifies a link to a long description of the image. This description should supplement the short description provided using the alt attribute.	RTExpr

Attribute Name	Description	Notes
name	The name of the DisplayField peer. This peer must be a child of the current parent ContainerView, TiledView, or ViewBean. The DisplayField name is resolved relative to the current parent view. The view peer may be any subtype of <JATO package>.view.DisplayField This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a ContainerView or a derivative of ContainerView (such as TiledView). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the ViewBean). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
normalize	Valid values: "true" or "false" Specifies that the src URL should be automatically normalized. This means that an absolute URL value will have the current servlet context prepended. A relative URL value will not have current servlet context prepended. Both relative and absolute src URL values will also be URL encoded. If not specified, "false" is assumed.	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
title	Offers advisory information about this element.	RTExpr
useMap	Associates an image map with an element. The image map is defined by a MAP element. The value of usemap must match the value of the name attribute of the associated MAP element.	RTExpr
vspace	Specifies the amount of white space to be inserted above and below the IMG. The default value is not specified, but is generally a small, non-zero length.	RTExpr
width	Image width override. When specified, the width and height attributes tell user agents to override the natural image or object size in favor of these values.	RTExpr
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 19

listbox

<jato:listbox>

Renders an HTML list as comprised of an HTML <input type="select"> element containing an arbitrary number of HTML <option> elements. This tag offers the convenience of treating the list as a single element and the flexibility of dynamic list content generation.

This tag is only valid when enclosed by both an HTML <form> element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:listbox name="itemSize" size="4" multiple="false"/>
```

The above would be rendered into the following HTML:

```
<select name="PageFoo.itemSize" size="4">
  <option value="S">Small</option>
  <option value="M">Medium</option>
  <option value="L">Large</option>
  <option value="XL">Extra Large</option>
</select>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
multiple	Valid values: "true" or "false" A value of "true" specifies that multiple items can be selected by the user. A value of "false" specifies that only a single item can be selected by the user. If not specified, "false" is assumed.	
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.html.SelectableGroup` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req

Attribute Name	Description	Notes
size	Valid values: positive number Specifies the number of rows in the list that should be visible at the same time. If not specified, the size attribute is omitted from the rendered HTML	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 20

password

<jato:password>

Renders an HTML `<input type="password">` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:password name="newPassword" size="8" maxLength="16"/>
```

The above would be rendered into the following HTML:

```
<input type="password" name="PageFoo.newPassword" value=""
maxLength="16" size="8">
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a password tag, "value" refers to the value that will be rendered in the HTML password element's "value" attribute	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
maxLength	Valid values: positive number Specifies the maximum number of characters the user may enter in the field. If not specified, an unlimited number of characters may be entered into the field.	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req

Attribute Name	Description	Notes
readOnly	Valid values: "true" or "false" A value of "true" specifies that the text contained within this element cannot be changed by the user. If not specified, "false" is assumed.	RTExpr
size	Valid values: positive number Specifies the initial width of the control in number of characters. If not specified, the size attribute is omitted from the rendered HTML	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 21

radioButtons

<jato:radioButtons>

Renders a group of HTML radio buttons as comprised of an arbitrary number of HTML `<input type="radio">` elements which share the same name attribute value. This tag offers the convenience of treating the set of radio buttons as a single element and the flexibility of dynamic choice generation.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:radioButtons name="incomeRange" layout="vertical"/>
```

The above would be rendered into the following HTML:

```
<input type="radio" name="PageFoo.incomeRange" value=""
checked>Nondisclosed

<br>

<input type="radio" name="PageFoo.incomeRange" value="19">$0
to $19,999

<Br>

<input type="radio" name="PageFoo.incomeRange" value=
"49">$20,000-$49,999

<Br>

<input type="radio" name="PageFoo.incomeRange" value=
"50">$50,000 or more

<Br>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
layout	Valid values: "h", "horizontal" or anything beginning with "h" A value beginning with "h" specifies a horizontal arrangement of radio buttons within this group. No value, or a value beginning with a letter other than "h" specifies a vertical alignment. The vertical arrangement is achieved by appending a `<Br>` element after each of the rendered HTML radio button controls.	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.html.SelectableGroup` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
style	CSS styles to be applied to this HTML element.	RTExpr

Attribute Name	Description	Notes
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

CHAPTER 22

text

<jato:text>

Renders a display field value as plain HTML body text. This tag is only valid when nested inside of a useViewBean or containerView or tiledView tag body.

This tag does not require an enclosing `<form>` element but must be enclosed by a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:text name="ssn" formatType="string" formatMask="???-??-
????"/>
```

The above would be rendered into the following HTML:

```
123-45-6789
```

Attribute Name	Description	Notes
defaultValue	The value to render if the `DisplayField` view peer's value is `null`.	RTExpr
escape	Valid values: "true" or "false" A value of "true" specifies that the value be HTML escaped. If not specified, a value of "true" is assumed.	
formatMask	The valid values for the `formatMask` attribute depend on the value of `formatType` attribute. For `formatType` values of "string" or "alpha": The format mask consists of any arbitrary combination of static text and the reserved metacharacter "?". Each "?" metacharacter serves as a placeholder for a single character in the value being formatted, proceeding from left to right. A literal "?" can be specified by escaping the "?" character with "\\". For `formatType` values of "dec" or "curr": The format mask consists of two patterns separated by a semicolon: a primary pattern (specifies precision and thousandths separator) and a negative number pattern (see `java.text.DecimalFormat` for pattern details). (*See table below.)	

Attribute Name	Description	Notes
formatType	Valid values: "string" (also "alpha"), "DEC", or "curr", or "date" Each type has specific masking rules (see the `formatMask` attribute for details). A value of "string" supports the formatting of arbitrary text via the application of a simple text mask. This allows dynamic content to be arbitrarily combined with static text. A value of "alpha" is identical to "string". A value of "DEC" supports the formatting of numeric values via the application of a numeric mask. A value of "curr" supports the formatting of currency values via the application of a currency mask. The currency format should simply take a decimal specification and apply the locale specific currency format. Currently, currency formatting is not working due to an unresolved bug. A value of "date", although legal, at present has no supporting date formatting implementation, due to the fact that the `java.text.DateFormat` class does not lend itself easily to the generic formatting support that a taglib implementation would require. We are investigating ways around this limitation. Therefore, it is suggested that developers wishing to apply date formatting implement the `begin<fieldName>Display(ChildDisplayEvent event)` and/or `end<fieldName>Display(ChildContentDisplayEvent event)` events and therein take explicit advantage of the date formatting support available in the `java.text.DateFormat` class. The JATO implementation of text formatting relies heavily on the `java.text` package and the format related classes therein. Please see that package for more information. See also the `<JATO package>.util.HtmlUtil` class.	
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
	This tag does not support any JavaScript events.	

(*See formatMask above.)

Format Type	Format Mask	Sample Value	Rendered Value
string	???-??-????	123456789	123-45-6789
string	??HELLO??	abcd	abHELLOcd
string	??HELLO\\???	abcd	abHELLO?cd
DEC	###0; (-#)	1000000.99	1000001
DEC	#,##0.00; (-#)	1000000.99	1,000,000.99
DEC	###0.00; (-#)	1000000.99	1000000.99
DEC	###0; (-#)	-1000000.99	(-1000001 BUG - Should be (-1000001)
DEC	###0; (#)	-1000000.99	(1000001 BUG - Should be (1000001)
DEC	#,##0.00; -#	-1000000.99	-1,000,000.99

CHAPTER 23

textArea

<jato:textArea>

Renders an HTML `<textarea>` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:textArea name="comments" cols="80" rows="5" wrap=
"virtual"/>
```

The above would be rendered into the following HTML:

```
<textarea name="PageFoo.comments" cols="80" rows="5" wrap=
"virtual">

...

</textarea>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
cols	Valid values: positive number Specifies the visible width of the field in average character widths.	Req, RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a textArea tag, "value" refers to the text that will be rendered between the opening and closing tag of the textarea element	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of the opening `<textarea>` element	RTExpr
formatMask	The valid values for the `formatMask` attribute depend on the value of `formatType` attribute. For `formatType` values of "string" or "alpha": The format mask consists of any arbitrary combination of static text and the reserved metacharacter "?". Each "?" metacharacter serves as a placeholder for a single character in the value being formatted, proceeding from left to right. A literal "?" can be specified by escaping the "?" character with "\\". For `formatType` values of "DEC" or "curr": The format mask consists of two patterns separated by a semicolon: a primary pattern (specifies precision and thousandths separator) and a negative number pattern (see `java.text.DecimalFormat` for pattern details). (*See table below.)	

Attribute Name	Description	Notes
formatType	Valid values: "string" (also "alpha"), "DEC", or "curr", or "date" Each type has specific masking rules (see the `formatMask` attribute for details). A value of "string" supports the formatting of arbitrary text via the application of a simple text mask. This allows dynamic content to be arbitrarily combined with static text. A value of "alpha" is identical to "string". A value of "DEC" supports the formatting of numeric values via the application of a numeric mask. A value of "curr" supports the formatting of currency values via the application of a currency mask. The currency format should simply take a decimal specification and apply the locale specific currency format. Currently, currency formatting is not working due to an unresolved bug. A value of "date", although legal, at present has no supporting date formatting implementation, due to the fact that the `java.text.DateFormat` class does not lend itself easily to the generic formatting support that a taglib implementation would require. We are investigating ways around this limitation. Therefore, it is suggested that developers wishing to apply date formatting implement the `begin<fieldName>Display(ChildDisplayEvent event)` and/or `end<fieldName>Display(ChildContentDisplayEvent event)` events and therein take explicit advantage of the date formatting support available in the `java.text.DateFormat` class. The JATO implementation of text formatting relies heavily on the `java.text` package and the format related classes therein. Please see that package for more information. See also the `<JATO package>.util.HtmlUtil` class.	
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
readOnly	Valid values: "true" or "false" A value of "true" specifies that the text contained within this element cannot be changed by the user. If not specified, "false" is assumed.	RTExpr

Attribute Name	Description	Notes
rows	Valid values: positive number Specifies the number of visible text lines in the rendered control	Req, RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
wrap	Valid values: valid wrap type supported by the browser, such as "virtual" or "physical" Specifies the value that will be rendered in the HTML textarea element's "wrap" attribute. If not specified, the wrap attribute is omitted from the rendered HTML	RTExpr
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

(*See formatMask above.)

Format Type	Format Mask	Sample Value	Rendered Value
string	???-??-????	123456789	123-45-6789
string	??HELLO??	abcd	abHELLOcd
string	??HELLO\\???	abcd	abHELLO?cd
DEC	###0; (-#)	1000000.99	1000001
DEC	#,##0.00; (-#)	1000000.99	1,000,000.99
DEC	###0.00; (-#)	1000000.99	1000000.99
DEC	###0; (-#)	-1000000.99	(-1000001 BUG - Should be (-1000001)
DEC	###0; (#)	-1000000.99	(1000001 BUG - Should be (1000001)
DEC	#,##0.00; -#	-1000000.99	-1,000,000.99

CHAPTER 24

textField

<jato:textField>

Renders an HTML `<input type="text">` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean, containerView, or tiledView tag. This tag may not contain any body content.

Example:

```
<jato:textField name="firstName" size="12" maxLength="32"/>
```

The above would be rendered into the following HTML:

```
<input type="text" name="PageFoo.firstName" value="John"
maxLength="32" size="12">
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a textField tag, "value" refers to the value that will be rendered in the HTML text element's "value" attribute	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
formatMask	The valid values for the `formatMask` attribute depend on the value of `formatType` attribute. For `formatType` values of "string" or "alpha": The format mask consists of any arbitrary combination of static text and the reserved metacharacter "?". Each "?" metacharacter serves as a placeholder for a single character in the value being formatted, proceeding from left to right. A literal "?" can be specified by escaping the "?" character with "\\". For `formatType` values of "DEC" or "curr": The format mask consists of two patterns separated by a semicolon: a primary pattern (specifies precision and thousandths separator) and a negative number pattern (see `java.text.DecimalFormat` for pattern details). (*See table below.)	

Attribute Name	Description	Notes
formatType	Valid values: "string" (also "alpha"), "DEC", or "curr", or "date" Each type has specific masking rules (see the `formatMask` attribute for details). A value of "string" supports the formatting of arbitrary text via the application of a simple text mask. This allows dynamic content to be arbitrarily combined with static text. A value of "alpha" is identical to "string". A value of "DEC" supports the formatting of numeric values via the application of a numeric mask. A value of "curr" supports the formatting of currency values via the application of a currency mask. The currency format should simply take a decimal specification and apply the locale specific currency format. Currently, currency formatting is not working due to an unresolved bug. A value of "date", although legal, at present has no supporting date formatting implementation, due to the fact that the `java.text.DateFormat` class does not lend itself easily to the generic formatting support that a taglib implementation would require. We are investigating ways around this limitation. Therefore, it is suggested that developers wishing to apply date formatting implement the `begin<fieldName>Display(ChildDisplayEvent event)` and/or `end<fieldName>Display(ChildContentDisplayEvent event)` events and therein take explicit advantage of the date formatting support available in the `java.text.DateFormat` class. The JATO implementation of text formatting relies heavily on the `java.text` package and the format related classes therein. Please see that package for more information. See also the `<JATO package>.util.HtmlUtil` class.	
maxLength	Valid values: positive number Specifies the maximum number of characters the user may enter in the field. If not specified, an unlimited number of characters may be entered into the field.	RTExpr
name	The name of the `DisplayField` peer. This peer must be a child of the current parent `ContainerView`, `TiledView`, or `ViewBean`. The `DisplayField` name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView` (such as `TiledView`). Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req

Attribute Name	Description	Notes
readOnly	Valid values: "true" or "false" A value of "true" specifies that the text contained within this element cannot be changed by the user. If not specified, "false" is assumed.	RTExpr
size	Valid values: positive number Specifies the initial width of the control in number of characters. If not specified, the size attribute is omitted from the rendered HTML	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767. **This tag also supports the following JavaScript events:** onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

(*See formatMask above.)

Format Type	Format Mask	Sample Value	Rendered Value
string	???-??-????	123456789	123-45-6789
string	??HELLO??	abcd	abHELLOcd
string	??HELLO\\???	abcd	abHELLO?cd
DEC	###0; (-#)	1000000.99	1000001
DEC	#,##0.00; (-#)	1000000.99	1,000,000.99
DEC	###0.00; (-#)	1000000.99	1000000.99
DEC	###0; (-#)	-1000000.99	(-1000001 BUG - Should be (-1000001)
DEC	###0; (#)	-1000000.99	(1000001 BUG - Should be (1000001)
DEC	#,##0.00; -#	-1000000.99	-1,000,000.99

CHAPTER 25

validatingTextArea

<jato:validatingTextArea>

Renders an HTML `<textarea>` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean or containerView tag. This tag may not contain any body content and may not appear within a tiledView tag.

The type of this tag's peer component must be `ValidatingDisplayField`.

Example:

```
<jato:validatingTextArea name="comments" cols="80" rows="5"
wrap="virtual"/>
```

The above would be rendered into the following HTML:

```
<textarea name="PageFoo.comments" cols="80" rows="5" wrap=
"virtual">

...

</textarea>
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
cols	Valid values: positive number Specifies the visible width of the field in average character widths.	Req, RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a validatingTextArea tag, "value" refers to the text that will be rendered between the opening and closing tag of the textarea element	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of the opening `<textarea>` element	RTExpr
formatMask	The valid values for the `formatMask` attribute depend on the value of `formatType` attribute. For `formatType` values of "string" or "alpha": The format mask consists of any arbitrary combination of static text and the reserved metacharacter "?". Each "?" metacharacter serves as a placeholder for a single character in the value being formatted, proceeding from left to right. A literal "?" can be specified by escaping the "?" character with "\\". For `formatType` values of "DEC" or "curr": The format mask consists of two patterns separated by a semicolon: a primary pattern (specifies precision and thousandths separator) and a negative number pattern (see `java.text.DecimalFormat` for pattern details). (*See table below.)	

Attribute Name	Description	Notes
formatType	Valid values: "string" (also "alpha"), "DEC", or "curr", or "date" Each type has specific masking rules (see the `formatMask` attribute for details). A value of "string" supports the formatting of arbitrary text via the application of a simple text mask. This allows dynamic content to be arbitrarily combined with static text. A value of "alpha" is identical to "string". A value of "DEC" supports the formatting of numeric values via the application of a numeric mask. A value of "curr" supports the formatting of currency values via the application of a currency mask. The currency format should simply take a decimal specification and apply the locale specific currency format. Currently, currency formatting is not working due to an unresolved bug. A value of "date", although legal, at present has no supporting date formatting implementation, due to the fact that the `java.text.DateFormat` class does not lend itself easily to the generic formatting support that a taglib implementation would require. We are investigating ways around this limitation. Therefore, it is suggested that developers wishing to apply date formatting implement the `begin<fieldName>Display(ChildDisplayEvent event)` and/or `end<fieldName>Display(ChildContentDisplayEvent event)` events and therein take explicit advantage of the date formatting support available in the `java.text.DateFormat` class. The JATO implementation of text formatting relies heavily on the `java.text` package and the format related classes therein. Please see that package for more information. See also the `<JATO package>.util.HtmlUtil` class.	
name	The name of the `ValidatingDisplayField` peer. This peer must be a child of the current parent `ContainerView` or `ViewBean`. The name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView`. Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req
readOnly	Valid values: "true" or "false" A value of "true" specifies that the text contained within this element cannot be changed by the user. If not specified, "false" is assumed.	RTExpr

Attribute Name	Description	Notes
rows	Valid values: positive number Specifies the number of visible text lines in the rendered control	Req, RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS styles to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
wrap	Valid values: valid wrap type supported by the browser, such as "virtual" or "physical" Specifies the value that will be rendered in the HTML textarea element's "wrap" attribute. If not specified, the wrap attribute is omitted from the rendered HTML	RTExpr
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

(*See formatMask above.)

Format Type	Format Mask	Sample Value	Rendered Value
string	???-??-????	123456789	123-45-6789
string	??HELLO??	abcd	abHELLOcd
string	??HELLO\\???	abcd	abHELLO?cd
DEC	###0; (-#)	1000000.99	1000001
DEC	#,##0.00; (-#)	1000000.99	1,000,000.99
DEC	###0.00; (-#)	1000000.99	1000000.99
DEC	###0; (-#)	-1000000.99	(-1000001 BUG - Should be (-1000001)
DEC	###0; (#)	-1000000.99	(1000001 BUG - Should be (1000001)
DEC	#,##0.00; -#	-1000000.99	-1,000,000.99

CHAPTER 26

validatingTextField

<jato:validatingTextField>

Renders an HTML `<input type="text">` element.

This tag is only valid when enclosed by both an HTML `<form>` element and a useViewBean or containerView tag. This tag may not contain any body content, and may not be used within a tiledView tag.

The type of this tag's peer component must be `ValidatingDisplayField`.

Example:

```
<jato:validatingTextField name="firstName" size="12"
maxLength="32"/>
```

The above would be rendered into the following HTML:

```
<input type="text" name="PageFoo.firstName" value="John"
maxLength="32" size="12">
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
defaultValue	The value to use, if the `DisplayField` view peer's value is `null`. For a validatingTextField tag, "value" refers to the value that will be rendered in the HTML text element's "value" attribute	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
formatMask	The valid values for the `formatMask` attribute depend on the value of `formatType` attribute. For `formatType` values of "string" or "alpha": The format mask consists of any arbitrary combination of static text and the reserved metacharacter "?". Each "?" metacharacter serves as a placeholder for a single character in the value being formatted, proceeding from left to right. A literal "?" can be specified by escaping the "?" character with "\\". For `formatType` values of "DEC" or "curr": The format mask consists of two patterns separated by a semicolon: a primary pattern (specifies precision and thousandths separator) and a negative number pattern (see `java.text.DecimalFormat` for pattern details). (*See table below.)	

Attribute Name	Description	Notes
formatType	Valid values: "string" (also "alpha"), "DEC", or "curr", or "date" Each type has specific masking rules (see the `formatMask` attribute for details). A value of "string" supports the formatting of arbitrary text via the application of a simple text mask. This allows dynamic content to be arbitrarily combined with static text. A value of "alpha" is identical to "string". A value of "DEC" supports the formatting of numeric values via the application of a numeric mask. A value of "curr" supports the formatting of currency values via the application of a currency mask. The currency format should simply take a decimal specification and apply the locale specific currency format. Currently, currency formatting is not working due to an unresolved bug. A value of "date", although legal, at present has no supporting date formatting implementation, due to the fact that the `java.text.DateFormat` class does not lend itself easily to the generic formatting support that a taglib implementation would require. We are investigating ways around this limitation. Therefore, it is suggested that developers wishing to apply date formatting implement the `begin<fieldName>Display(ChildDisplayEvent event)` and/or `end<fieldName>Display(ChildContentDisplayEvent event)` events and therein take explicit advantage of the date formatting support available in the `java.text.DateFormat` class. The JATO implementation of text formatting relies heavily on the `java.text` package and the format related classes therein. Please see that package for more information. See also the `<JATO package>.util.HtmlUtil` class.	
maxLength	Valid values: positive number Specifies the maximum number of characters the user may enter in the field. If not specified, an unlimited number of characters may be entered into the field.	RTExpr
name	The name of the `ValidatingDisplayField` peer. This peer must be a child of the current parent `ContainerView` or `ViewBean`. The name is resolved relative to the current parent view. The view peer must be of type `<JATO package>.view.DisplayField` This name may be a qualified view path, using forward slashes ("/") as delimiters. All components in the path except the last must refer to a `ContainerView` or a derivative of `ContainerView`. Both relative and absolute paths are possible. If a name path begins with a forward slash, the name is assumed to be relative to the root view (the `ViewBean`). If the path does not begin with a forward slash, the name is assumed to refer to a child relative to the current container. Two dots ("..") may be used to refer to the container that is the parent of the current container. Examples: `/header/orderList/customerName` (absolute from root view) `orderList/customerName` (relative to current container) `../footer/orderList/customerName` (relative to parent)	Req

Attribute Name	Description	Notes
readOnly	Valid values: "true" or "false" A value of "true" specifies that the text contained within this element cannot be changed by the user. If not specified, "false" is assumed.	RTExpr
size	Valid values: positive number Specifies the initial width of the control in number of characters. If not specified, the size attribute is omitted from the rendered HTML	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767. **This tag also supports the following JavaScript events:** onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

(*See formatMask above.)

Format Type	Format Mask	Sample Value	Rendered Value
string	???-??-????	123456789	123-45-6789
string	??HELLO??	abcd	abHELLOcd
string	??HELLO\\???	abcd	abHELLO?cd
DEC	###0; (-#)	1000000.99	1000001
DEC	#,##0.00; (-#)	1000000.99	1,000,000.99
DEC	###0.00; (-#)	1000000.99	1000000.99
DEC	###0; (-#)	-1000000.99	(-1000001 BUG - Should be (-1000001)
DEC	###0; (#)	-1000000.99	(1000001 BUG - Should be (1000001)
DEC	#,##0.00; -#	-1000000.99	-1,000,000.99

CHAPTER 27

content

<jato:content>

Denotes a section of JSP content that has `begin<name>Display()` and `end<name>Display()` events associated with it. The association of these events works in the same way as for visual tags, except that there need be no peer by the specified name in the parent container view.

The entire body content of this tag may be conditionally suppressed by returning a value of false from the corresponding `begin<name>Display()` event. If the body content is not suppressed, the processed body content will be provided to the `end<name>Display()` event. The body content will be fully processed, including any embedded JSP or JATO tags, before being passed to this event.

Unlike the firing of display events for the visual tags, which may be suppressed by a combination of attributes on the tag itself or its parent tag, the firing of display events for this tag always occurs. Therefore, for efficiency reasons, developers should always implement at least one of the display events for a content tag—if no events are implemented, there is no reason to use the tag.

This tag is only valid when enclosed by a useViewBean, containerView, or tiledView tag.

Example:

```
...
Username: <jato:textField name="username"/>
<jato:content name="superuser">
<br>
Password: <jato:textField name="password"/>
```

```
</jato:content>

<hr>

...
```

The above would be rendered into the following HTML if and only if the `beginSuperuserDisplay()` event in the current parent view returned true:

```
...

Username: <input type="text" name="PageFoo.username" value=
"scott">

<br>

Password: <input type="text" name="PageFoo.password" value=
"tiger">

<hr>

...
```

The `endSuperuserDisplay()` event would receive the following HTML as a method parameter:

```
<br>

Password: <input type="text" name="PageFoo.password" value=
"tiger">
```

If the `beginSuperuserDisplay()` event instead returned false, the following would be rendered:

```
...

Username: <input type="text" name="PageFoo.username" value=
"scott">

<hr>

...
```

The `endSuperuserDisplay()` event would not be invoked.

Attribute Name	**Description**	**Notes**
name	The name of the content section, used to fire display events in the current parent view.	Req

CHAPTER **28**

pagelet

<jato:pagelet>

Allows JATO tags in an included JSP fragment to inherit the enclosing page's container view scope, and thus be included at request-time.

This tag is only valid when used inside an included JSP fragment (also known as a *pagelet*). The **`pagelet`** tag should enclose all other JATO tags in the pagelet.

Example:

`EnclosingPage.jsp`

```
<%@page info="E0130" language="java"%>
<%@taglib uri="/WEB-INF/jato.tld" prefix="jato"%>

<jato:useViewBean ...>
<jato:containerView name="foo">
  ...
  <jsp:include page="MyPagelet.jsp"/>
  ...
  </jato:containerView>
</jato:useViewBean>
```

`MyPagelet.jsp`

```
<%@page info="MyPagelet" language="java"%>
<%@taglib uri="/WEB-INF/jato.tld" prefix="jato"%>

<jato:pagelet>
  <jato:combobox name="month"/>...
</jato:pagelet>
```

CHAPTER 29

treeNode

<jato:treeNode>

Denotes an arbitratry JSP content section that will be rendered for any matching tree nodes. There may be more than one treeNode tag contained in a treeView tag. With each cycle of the TreeView traversal, any contained treeNode tags are evaluated for a match against the current node. If the treeNode tag matches the current node, then the content of the treeNode tag is rendered.

A treeNode tag is said to "match" the current TreeView node if all of the following is true (each comparison is considered true if the tag attribute is unspecified).

- Current node name matches tag attribute "nodeName"
- Current node type matches tag attribute "nodeType"
- Current node children matches tag attribute "hasChildren"
- Current node expanded state matches tag attribute "isExpanded"

Note that the nodeName and nodeType attributes support comma delimited lists of potential match criteria. The expanded state of the current node is determined by end user interaction with the **`treeNodeHandle`** tags associated with the current node.

This tag is only valid when ultimately enclosed by a **`treeView`** tag; it does not have to be a direct child tag of the enclosing treeView tag, and can in fact be arbitrarily nested below the **`treeView`** parent tag, even within other **`treeNode`** tags.

Example:

```
...
<jato:treeView name="treeMenu">
```

```
        <jato:treeNode nodeName="menuCategory" hasChildren="true" >
          <jato:href name="category"/>
        </jato:treeNode>

        <jato:treeNode nodeName="menuCategory" hasChildren="false"
        >
          <jato:text name="category"/>
        </jato:treeNode>

        <jato:treeNode nodeName="menuChoice,menuItem">
          <jato:href name="item"/>
        </jato:treeNode>

    </jato:treeView>
    ...
```

Attribute Name	Description	Notes
nodeName	Valid values: Arbitrary string of comma delimited node names. Specifies a match on current node name. The attribute may specify a comma delimited list of node names with the intent of matching the current node against any one of the specified names. If not specified, this does not factor in node match evaluation.	RTExpr
nodeType	Valid values: Arbitrary string of comma delimited node types. Specifies a match on current node type. The attribute may specify a comma delimited list of node types with the intent of matching the current node against any one of the specified types. If not specified, this does not factor in node match evaluation.	RTExpr
hasChildren	Valid values: "true" or "false" A value of "true" specifies that the current tag should only match the current node if the current node has children. If not specified, this does not factor in node match evaluation.	RTExpr
isExpanded	Valid values: "true" or "false" A value of "true" specifies that the current tag should only match the current node if the current node is already "expanded". If not specified, this does not factor in node match evaluation.	RTExpr

CHAPTER 30

treeLevel

<jato:treeLevel>

Denotes an arbitratry JSP content section that will be rendered "level times" for matching node levels. There may be more than one treeLevel tag contained in a treeView tag. With each cycle of the TreeView traversal, any contained treeLevel tags are evaluated for a match with the current node depth level. If the treeLevel tag matches the current node depth level then the content of the tag is rendered a number of times equal to the current level.

A treeLevel tag is said to "match" the current level if all of the following is true (each comparison is considered true if the tag attribute is unspecified):

- Current level >= tag attribute "minLevel"
- Current level <= tag attribute "maxLevel"

If the treeLevel tag matches the current tree depth level, then the body content will be processed a number of times equal to the current level, including any embedded JSP or JATO tags. For example, if the current node level is 4, then the body content of any matching treeLevel tags would be rendered 4 times for the current node. Typically, one uses treeLevel tags to provide level-specific indentation or other structural markup.

This tag is only valid when ultimately enclosed by a **`treeView`** tag; it does not have to be a direct child tag of the enclosing **`treeView`** tag, and can in fact be arbitrarily nested below the treeView parent tag.

Example:

```
<jato:treeView name="treeMenu">
```

```
<!-- Add a space for each level of the current node -->

<jato:treeLevel>

&nbsp;

</jato:treeLevel>


...


</jato:treeView>
```

Attribute Name	Description	Notes
minLevel	Valid values: positive number Specifies the minimum node depth level at which to match.	RTExpr
maxLevel	Valid values: positive number Specifies the maximum node depth level at which to match.	RTExpr
offset	Valid values: positive or negative integer Specifies a positive or negative node level depth modifier to apply when rendering this tag. An offset of -*n* will cause the tag to render (level-*n*) times, whereas an offset of n will cause the tag to render (level+*n*) times.	RTExpr

CHAPTER 31

treeNodeHandle

<jato:treeNodeHandle>

Denotes a visual control that end users can select to toggle the expanded/collapsed state of a particular tree node. This tag will be rendered as an HTML `<a href= "..." >...</a>` element, with an implicit URL generated by the tag and handled by the parent tree view automatically when activated.

This tag may contain body content. This content will be included between the `<a>` and `</a>` tags of the rendered element and thus appear as the visible, clickable HREF element.

This tag is only valid when ultimately enclosed by a treeView tag. This tag may be directly parented by a treeNode tag.

Example:

```
...


<jato:treeNodeHandle>
  <img src="handle.gif">
</jato:treeNodeHandle>


...
```

Attribute Name	Description	Notes
accessKey	Assigns an access key to the rendered HTML element. In general, the access key should be included in label text or wherever the access key is to apply.	RTExpr
disabled	Valid values: "true" or "false" Disables the HTML control for user input. • Disabled controls do not receive focus • Disabled controls are skipped in tabbing navigation If not specified, "false" is assumed.	RTExpr
elementId	Assigns a name to an element. This name must be unique in an HTML document. *Please note, this corresponds to the HTML "id" attribute. It has no specific meaning to the JATO framework.* The id attribute has several roles in HTML: • As a style sheet selector • As a target anchor for hypertext links • As a means to reference a particular element from a script	RTExpr
extraHtml	Arbitrary HTML that will be rendered just prior to the closing ">" of this HTML element	RTExpr
queryParams	Specifies an arbitrary number of name value pairs which will be appended to the HREF's query string. The name value pairs should follow the format "name=value", where value is URL encoded by the JSP author as needed. Additional name value pairs must be delimited by the '&' character. Example: queryParams="fname=Mike&lname=Jones"	RTExpr
style	CSS styles to be applied to this HTML element.	RTExpr
styleClass	CSS stylesheet class to be applied to this HTML element.	RTExpr
tabIndex	Specifies the position of the current element in the tabbing order for the current document. This value must be a number between 0 and 32767.	
title	Offers advisory information about this element.	RTExpr
trim	Valid values: "true" or "false" A value of "true" specifies that the tag content should be trimmed of all leading and trailing whitespace. A value of "false" preserves the whitespace. If not specified, "true" is assumed.	
	This tag also supports the following JavaScript events: onBlur, onChange, onClick, onDblClick, onFocus, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onSelect	

Index