

# HTLM and XHTML Support

## Table of contents

1 Introduction.....	2
2 Basic Tag Structure.....	2
2.1 Html Tag.....	2
2.2 Body Tag.....	3
2.3 ScriptBlock Tag.....	3
3 Specifying the HTML Format.....	3
3.1 Format Identifiers.....	3
3.2 Setting the WebApp Format.....	3
3.3 Setting the Document Format.....	4

## 1. Introduction

The following topic explains the support for creating valid XHTML and HTML from the NetUI tags. The NetUI tags currently support three levels of the XHTML and HTML specs, HTML 4.01 Loose (Transitional DTD), HTML 4.01 Loose Quirks mode and XHTML 1.0 Transitional. By default, the tags support HTML 4.01 Loose, Quirks mode.

Future versions of NetUI may support HTML 4.01 Strict and XHTML Strict. This is currently not supported because there are a number of visual presentation attributes supported by the NetUI HTML tags, which are not supported by HTML 4.01 Strict DTD (for example `<netui:body>`'s `bgcolor` attribute). In order to support these strict modes, the NetUI tags would need to report runtime errors when the deprecated attributes are set.

## 2. Basic Tag Structure

The minimum JSP should be of the following structure:

### Basic JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"%>
<%@ taglib prefix="netui"
uri="http://beehive.apache.org/netui/tags-html-1.0"%>
<netui:html>
  <head>
    <title>Page Title</title>
    <netui:base/>
  </head>
  <netui:body>
  </netui:body>
</netui:html>
```

There are two primary tags that create the valid HTML structure and provide the core services of the NetUI tag library, `<netui:html>` and `<netui:body>`. These two tags are related to each other. Together they work together to create valid HTML. The page above represents the typical minimum page. The **html** tag will output the HTML `html` element. The `<head>` contains a title and the `<netui:base>` tag which is used to locate resources and links on the page. The **body** tag will then output the HTML body element.

### 2.1. Html Tag

The `<netui:html>` JSP tag typically acts as a top level container and in this role is responsible for rendering the result of many of the core services of the tags. These services include gathering up JavaScript and runtime errors and then renders them at the end of the page. For more information on the Html tag see the [ScriptContainers](#) topic.

Typically if the `html` tag rendered this content it would be illegal because the end tag `</body>` immediately precedes the `</html>` end tag and there is no legal content allowed between these two end tags. In order to create legal HTML, the `<netui:body>` tag defers to the `Html` tag to render content to the generated page before the body end tag is rendered. If the `<netui:body>` tag is not present, then the `Html` tag will render JavaScript and any runtime errors right before the `</html>` end tag.

## 2.2. Body Tag

The `<netui:body>` JSP tag will output the HTML body tag. In addition, it works with the `<netui:html>` tag to render the JavaScript and runtime errors before the `</body>` end tag. In addition, it will also work with the `<netui:scriptBlock>` tag to position any JavaScript that must be positioned either before or after the framework generated JavaScript. For more information see the [ScriptBlock](#) information in the JavaScript Support topic.

## 2.3. ScriptBlock Tag

The primary role of the `<netui:scriptBlock>` is to render HTML `<script>` tags and position JavaScript on the page in relationship to the framework generated JavaScript. The `ScriptBlock` requires both the `<netui:html>` and `<netui:body>` tag to position the output. Without these two tags, the JavaScript is written out inline. For more information see the [ScriptBlock](#) information in the JavaScript Support topic.

## 3. Specifying the HTML Format

This section describes the methods for setting the default HTML format for a WebApp. Further, the format can be set on a JSP to override the WebApp default.

### 3.1. Format Identifiers

The `Html` tags currently support three HTML formats. These are identified by the following identifier values:

- `html4-loose` -- HTML 4.01 Transitional DTD (loose)
- `html4-loose-quirks` -- HTML 4.01 Transitional DTD, with Quirks mode turned on
- `xhtml1-transitional` -- XHTML 1.0 Transitional

The identifier values are used set the HTML format in both the tags and in the configuration file.

### 3.2. Setting the WebApp Format

By default, all WebApps use `html4-loose-quirks` as the format. In order to override the default format, you set the WebApp default in the `beehive-netui-config.xml`. The default is set by specifying the [doctype](#) value inside of the `jsp-tag-config` element. For information on the NetUI Configuration file see the [Reference Documentation: beehive-netui-config.xml File](#) topic.

In the following example, the default format for the WebApp is changed to the HTML 4.01 Transitional DTD. This value is found in the `beehive-netui-config.xml` file. The value of the `<doctype>` element must be one of the values specified above for the identifiers.

```
<jsp-tag-config>
  <doctype>html4-loose</doctype>
</jsp-tag-config>
```

### 3.3. Setting the Document Format

It is possible to override the HTML format on a document by document basis. This is done by setting the **documentType** of the `<netui:html>` tag to one of the identifiers above. In the example below, the output format for the document will be XHTML 1.0 Transitional. It will override the default value for the WebApp.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<netui:html documentType="xhtml1-transitional">
  ...
</netui:html>
```