UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA EN SISTEMAS INFORMÁTICOS
SISTEMAS Y PROCEDIMIENTOS (SYP 115)


# Manual de uso de etiquetas JSTL y EL

**Docentes:**
Ing. José Alberto Martínez Campos
Ing. Ricardo Amed Guardado Hernández

**Integrantes:**

| Apellidos | Nombres | Carné | GT |
|---|---|---|---|
| Hernández Mejía | Kenia | HM14005 | 1 |
| Herrera Jirón | Melvin Ulises | HJ13003 | 2 |
| Lovos Molina | Osiris Alexander | LM12013 | 1 |
| Morales Sánchez | Ilich Daniel Manrique | MS11053 | 2 |
| Navarro Navarro | Reynaldo Esaú | NN13002 | 2 |

**1 de abril de 2016**

# Contenido

Introducción ..... 1

Objetivos ..... 2

Alcances de la investigación ..... 3

Limitaciones ..... 3

Conclusiones ..... 32

Bibliografía ..... 33

# Introducción

El presente estudio explica detalladamente en que consiste cada una de las etiquetas del lenguaje JSP Standard Tag Library o mejor dicho Java Server Pages Standar Tag Library (Librería de Etiquetas Estándar de Páginas de Servidor Java) o llamado por su acrónimo JSTL, llamado así de ahora en adelante.

Es un entorno web en que se utilizan este tipo de librerías las cuales se pueden tomar como introducción a los diversos frameworks que posee el lenguaje Java. Entre estos, tenemos a JSF (Java Server Faces) el framework Spring MVC, NHibernate o las etiquetas PrimeFaces, entre otras.

La versión válida de JSTL que utilizamos es la versión 2.1 y encapsula las funciones centrales más comunes a muchas aplicaciones JSP. En lugar de combinar numerosas etiquetas de muchos vendedores en su aplicación JSP, JSTL le permite implementar un simple conjunto estándar de etiquetas. Esta estandarización le permite desplegar sus aplicaciones en cualquier contenido JSP que soporte JSTL y hace parecer que la implementación de la etiqueta está optimizada.

JSTL tiene etiquetas como iteraciones, condicionales, para control de flujo, etiquetas para controlar XML, documentos, etiquetas de internacionalización, etiquetas para acceder a bases de datos SQL y funciones comúnmente utilizadas.

# Objetivos

## Objetivo General

Instruir el uso de herramientas como son las etiquetas JSTL y EL para el desarrollo web con el fin de brindar enseñanza a los lectores y permitiendo desarrollarle nuevas opciones de trabajo.

## Objetivos específicos

- Brindar una base introductoria de las etiquetas JSTL y EL para su oportuno uso en los proyectos de desarrollo del lector.

- Demostrar la facilidad de uso de las etiquetas con sus correspondientes formas de uso.

- Guiar al lector a las alternativas que ofrece las etiquetas JSTL y el lenguaje de expresiones (EL) para expandir sus conocimientos.

## Alcances de la investigación

- El presente estudio explica detalladamente en que consiste cada una de las etiquetas del lenguaje JSTL, que proporciona cuatro bibliotecas de etiquetas, para el desarrollo de aplicaciones web dinámicas.

- La investigación abarca únicamente las 4 librerías de JSTL (Core, XML, SQL, FMT), además de EL (Expression Language), formando parte de los componentes de la tecnología Java EE.

- El manual provee un ejemplo de uso para cada una de las etiquetas que integran a las bibliotecas de JSTL.

## Limitaciones

- Como buena práctica no es recomendable utilizar las etiquetas de la librería SQL Tag Library dado que se pretende manejar todo en la capa de código.

- Las etiquetas no se puede predefinir dinámicamente con código, es decir, no se puede insertar etiquetas a través de programación.

## 1. Lenguaje De Expresiones <EL>

Antes de profundizar en las distintas áreas funcionales de la JSTL, debemos comenzar con el lenguaje de expresión. Esta es una de las características más importantes de JSTL y es una diferencia prominente de la especificación JSP 2.0. El lenguaje de expresión (EL) permite una sintaxis mucho más simple para hacer los datos de aplicaciones y la manipulación para el autor de la página. Actualmente EL en el JSTL sólo se puede utilizar con la etiqueta los valores de atributos, principalmente en acciones que residen en la biblioteca de etiquetas Core. Es posible utilizar EL dentro del texto de la plantilla si se está trabajando con la especificación JSP 2.0. Expresiones en el texto de la plantilla no son compatibles si está utilizando JSTL 1.0 con JSP 1.2

NOTA: Todas las funciones son tratadas como cadenas nulas como cadenas vacías.
Uso: ${fn:function(arg0, ...)}

| Función | Descripción |
|---|---|
| **fn:contains (string, subcadena) : boolean** | Devuelve verdadero si la subcadena está contenida en la cadena ; falso , de lo contrario |
| **fn:containsIgnoreCase (string,subcadena) : boolean** | Devuelve verdadero si la subcadena está contenida en la cadena independientemente del caso; falso, de otra manera. |
| **fn:endsWith (Cadena, sufijo ) : booleano** | Devuelve verdadero si la cadena termina con el sufijo especificado; falso, de lo contrario. |
| **fn:escapeXml (Cadena ) : String** | Escapes caracteres (por ejemplo cambiando " < " en "& lt ; " ) que pudiera ser interpretado como XML ( incluyendo HTML ) de marcado . |
| **fn:indexOf (Cadena, subcadena ) : int** | Devuelve un entero que representa el 0 índice basado en cadena de la primera ocurrencia de subcadena. Si es subcadena vacía, se devuelve 0 |
| **fn:join( Cadena [ ] , separador) : String** | Se une a todos los elementos de la matriz de cadenas en una sola cadena. El Separador aparta cada elemento de la resultante cadena. Si el separador es una cadena vacía, se unen los elementos sin un separador. |
| **f :Length (Colección o de cadena) : int** | Si se pasa una colección o matriz, el tamaño de la colección o matriz es devuelta; Si se pasa una cadena, el número de caracteres en la cadena es devuelto. |
| **fn:replace(Cadena de entrada, cadena Después, Cadena Antes) : String** | Sustituye en cadena de entrada, cada Ocurrencia de cadena antes con la cadena después. Se devuelve una cadena vacía si bien la cadena de entrada o la cadena antes están vacías. Si la cadena después está vacía, todas las apariciones de la Antes de cuerdas se retiran. |
| **fn:split(Cadena, delimitadores) :String[]** | Divide la cadena en una matriz de cadenas. Usando el conjunto dado de caracteres delimitadores. Los caracteres delimitadores no |

| | |
|---|---|
| | son incluidos en cualquier arreglo devuelto. |
| **fn:startsWith (Cadena, prefijo) : booleano** | Devuelve verdadero si la cadena se inicia con el prefijo especificado; falso, de lo contrario. Devuelve verdadero si el prefijo está vacío. |
| **fn:subString (Cadena, índice inicial , Índice de terminar ) :String** | Devuelve un subconjunto de cadena utilizando los índices de base cero - incluido el índice de empezar, pero excluyendo el índice final. |
| **fn:subcadena Después (cadena, subcadena):String** | Devuelve el subconjunto de cadena que le sigue la subcadena dada. |
| **fn:Antes de subcadena(Cadena, subcadena ):String** | Devuelve el subconjunto de cadena que procede la subcadena dada. |
| **fn:toLowerCase (Cadena ) : String** | Convierte todos los caracteres de una cadena a minúsculas. |
| **fn:toUpperCase (cadena) : String** | Convierte todos los caracteres de una cadena a mayúsculas. |
| **fn:trim (string) : String** | Elimina los espacios en blanco de ambos extremos de una cadena. |

Implicit Objects (maps)→ Objetos Implicitos (mapas).

Objetos implícitos disponibles en EL. Hay un buen número de objetos implícitos expuestos a través de EL. Estos objetos permiten para acceder a cualquier variable que se llevan a cabo en los ámbitos JSP particulares. Los objetos incluyen PageScope, requestScope, sessionScope, y applicationScope. Todos estos xScope objetos son mapas que en el mapa los ámbitos de aplicación respectivos nombres de atributo a sus valores. Utilizando los objetos implícitos param y paramValues, también es posible acceder a los parámetros de solicitud HTTP. Esto es válido para la información de cabecera solicitud, así como para el uso de los objetos implícitos. Encabezado y headerValues. Los objetos y param de cabecera son mapas que asignar el nombre de parámetro o de cabecera en una cadena. Esto es similar a hacer un ServletRequest.getParameter (String name) o ServletRequest.getHeader (String name). Los paramValues y headerValues son Mapas, ese parámetro mapa y los nombres de encabezado en una cadena [] de todos los valores de ese parámetro o encabezamiento. Una vez más, esto es como si se hubiera hecho ServletRequest.getParameterValues (String nombre) o ServletRequest.getHeaders (String) llama. El initParam da acceso a los parámetros de inicialización de contexto, mientras que la galleta expone las cookies recibidas en la solicitud. El pageContext objeto implícito da acceso a todos propiedades asociadas con el PageContext de una página JSP como el HttpServletRequest, ServletContext y objetos HttpSession y sus propiedades.

| Objetos | Descripción |
|---|---|
| **pageContext** | Página JSP objeto Contexto. |
| **pageScope** | Las variables de página con ámbito (válido sólo en un publicado en la página JSP). |
| **requestScope** | Solicitud de variables con ámbito (válido para un determinado solicitud). |
| **sessionScope** | Las variables de sesión con ámbito (válido para la sesión del usuario). |
| **applicationScope** | Las variables de ámbito de aplicación (válido para un dado contexto de |

| | |
|---|---|
| | aplicación). |
| **param** | Mapa del nombre de una orden de parámetro en una cadena el valor del parámetro. |
| **paramValues** | Mapa del nombre de una orden de parámetro en una cadena gama de valores de los parámetros. |
| **header** | Mapa del nombre de encabezado de solicitud a un colector valor de la cadena. |
| **headerValues** | Mapa del nombre de encabezado de solicitud de una cadena Conjunto de valores. |
| **cookie** | Mapa del nombre de la cookie a un objeto de la cookie. |
| **initParam** | Mapa del parámetro de inicialización de contexto<br>Nombrar a un valor de parámetro de cadena (establecido en web.xml). |

Veamos un par de muestras para impulsar el uso de objetos:

- $ { } pageContext.request.servletPath devolverá la ruta del servlet obtenido a partir de la HttpServletRequest.

- $ { } sessionScope.loginId devolverá el atributo ámbito de sesión llamado "loginid" o null si no se encuentra el atributo.

- $ { } param.bookId devolverá el valor de cadena del parámetro " bookId ", o null si se trata de extraviado.

- $ { } paramValues.bookId devolverá la cadena [ ] que contiene todos los valores de la bookId parámetro, o null si no se encuentra. Usando paramValues es particularmente útil si tener un formulario con casillas de verificación o por alguna otra razón podría tener un parámetro. Los valores múltiples como una caja de selección múltiple.

## Operadores Relacionales

Estos operadores son: ==,! = , < , > , < = , > = , Eq , ne , lt , gt , le, y GE . Los últimos seis operadores se ponen a disposición para evitar tener que utilizar referencias de entidad en la sintaxis XML. Las referencias de entidad a veces son necesarios, ya que, si se coloca un personaje como < dentro de un elemento XML, el analizador generará un error debido a su pensamiento que es el inicio de un nuevo elemento. Todos los caracteres XML ilegales tienen que ser sustituidos por referencias de entidad. Por ejemplo, si teniendo un elemento XML que se parecía a:

< Libro > si el precio < 50 entonces </Libro > tendríamos que sustituir el < & lt con lo que parecía: < Libro > si el precio & lt; 50 entonces </ Libro > y pudo analizar correctamente.

También se proporciona el operador de prefijo vacío para probar si un valor es nulo o vacío sentido " ". Por ejemplo: < c:if test = " $ { } " param.bookId vacío > Un libro debe ser seleccionado para procesar el pedido < / c:if >

| OPERADOR | DESCRIPCION |
|---|---|
| **== (eq)** | Igualdad. |
| **!= (ne)** | Desigualdad. |
| **< (lt)** | Menos que |
| **> (gt)** | Mayor que |
| **<= (le)** | Menor o igual que |
| **>= (ge)** | Mayor o igual que |

## Operadores aritméticos

Además de los operadores relacionales y el prefijo, también hay aritmética y lógica operadores que se pueden utilizar con EL. Los operadores aritméticos se componen de suma (+ ) , resta ( - ) , Multiplicación ( * ) , división ( / o div ) , y el resto / módulo (% o MOD ).

| OPERADOR | DESCRIPCION |
|---|---|
| **+** | Adicción |
| **-** | Sustracción |
| ***** | Multiplicación |
| **%(Mod)** | Cociente |
| **/** | División |

## Operadores Lógicos

| OPERADOR | DESCRIPCION |
|---|---|
| **&& (y)** | Verdadero si ambos operando son verdaderos; falso, de lo contrario. |
| **\|\| (o)** | Es cierto que si uno o ambos operando son verdaderos; falso, de otra manera. |
| **! (not)** | Es cierto que si el operando es falso ; falso , de lo contrario |

**Ejemplos:**

| Expresión | Descripción | Booleano |
|---|---|---|
| **${1 == 1}** | Es igual operador | Verdadero |
| **${1 != 1}** | No es igual operador | Falso |
| **${1 <= 1}** | Inferior o igual a igual | Verdadero |
| **${1 le 1}** | Inferior o igual a usar entidad referencia | Verdadero |
| **${1 == 1 \|\| 2 > 3}** | resultados de la comparación compuesto en ( Verdadero o falso ) | Verdadero |
| **${param.name == 'Sue'}** | Compruebe que el parámetro petición llamado nombre es igual a la cadena demandar. Observe el uso de comillas simples alrededor del valor. | Es cierto, porque nuestro JSP es estar llama con una cadena de consulta de operators.jsp ? name = Sue |
| **${empty param.name}** | Comprobar un valor de la solicitud parámetro llamado nombre. Si el valor es nulo o " " , se vaciará como resultado cierto | Falso, porque nuestro JSP es estar llama con una cadena de consulta de operators.jsp? name= Sue. |

## 2. Core Tag Library

El conjunto de etiquetas que están disponibles en la biblioteca de etiquetas Core entran en juego para probablemente casi cualquier cosa que va a hacer en las JSP. Vamos a caminar a través de ejemplos de código para ver cómo utilizamos cada una de las etiquetas proporcionadas en esta biblioteca.

Si se utilizará la librería CORE podemos importarla en la página que se necesitará, para ello sólo se debe de escribir el correspondiente taglib al inicio de la página JSP:

**<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>**

La zona núcleo comprende cuatro secciones funcionales distintas:

- Acciones de propósito general que se utilizan para manipular las variables de ámbito que podrían encontrarse dentro de una página JSP (Java Server Page). Estas acciones de propósito general también abarcan la gestión de errores.

- Acciones condicionales utilizados para hacer el procesamiento condicional dentro de un JSP.

- Acciones de iterador que hacen que sea fácil para iterar a través de colecciones de objetos.

- Las actividades de URL para hacer frente a los recursos de URL en un JSP.

Veamos cada sección funcional en la biblioteca de etiquetas Core un poco más de cerca.

### Acciones de uso general

Acciones para la representación de datos, creación y modificación de las variables con ámbito, y la captura de excepciones.

| **<c:out>** | | |
|---|---|---|
| **Descripción** | Evalúa una expresión y envía el resultado de la evaluación al JspWriter objeto actual. | |
| **Ejemplo de Uso** | **<c:out value=" Las muestras de referencia " escapeXml="falso" default="Muestras"/>** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **value**<br>**escapeXML** | Determina si escapeXml caracteres <,>, &, ', "en el resultando cadena debe ser convertir en sus correspondientes entidad de caracteres códigos. El valor por defecto es cierto. | Objeto booleano |
| **default** | El valor por defecto si, el valor resultante es nulo. | Objeto |

| <c:remove> | | |
|---|---|---|
| **Descripción** | Elimina una variable con ámbito. | |
| **Ejemplo de Uso** | **<c:remove var="miVariable" />** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **var** | Nombre restringido de la variable para ser eliminada. | Cadena |
| **scope** | Ámbito de var (variables). | Cadena |

| <c:set> | | |
|---|---|---|
| **Descripción** | Establece el valor de una variable con ámbito o una propiedad de un objeto de destino. | |
| **Ejemplo de Uso** | **<c:set value="<b> Datos de prueba </b>" var=" mis datos"/>** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **value var** | Nombre de las exportadas variables con ámbito para mantener el valor especificado en la acción.<br>El tipo de la variable con ámbito es cualquier tipo de valor de la expresión. | Objeto cadena |
| **scope target** | Cuyo objeto de destino se establecerá la propiedad. Debe evaluar a un JavaBeans objeto con la propiedad de la moda propiedad, o al java.util.Map objeto. | Objeto cadena |
| **property** | Nombre de la propiedad a ser establecido en el objeto de destino. | Cadena |

| <c:catch> | |
|---|---|
| **Descripción** | Catches a java.lang. Lanzada por cualquiera de sus acciones anidadas. |
| **Ejemplo de Uso** | **<c:catch var="error">**<br>**acciones anidadas**<br>**</c:catch>** |
| ***Atributos*** | |

| Nombre | Descripción | Tipo |
|---|---|---|
| **var** | Nombre de la variable para la excepción lanzada desde una acción jerarquizada. El tipo de la variable de ámbito es el tipo de excepción lanzado. | Cadena |

## Acciones condicionales

Acciones para marcado de procesamiento basado en lógica de condiciones.

| **<c:if>** | |
|---|---|
| **Descripción** | Evalúa su contenido cuerpo si la expresión especificada con la prueba atributo es cierto. |
| **Ejemplo de Uso** | **Con el contenido del cuerpo**<br>**<c:if prueba="${varA < 5}" >**<br>**Hacer algo**<br>**</c:if>**<br>**Sin contenido del cuerpo**<br>**<c:if prueba="${varA < 5}" var="resultado" />** |

***Atributos***

| Nombre | Descripción | Tipo |
|---|---|---|
| **test** | La condición de prueba que determina si el contenido del mensaje debe ser procesado. | Cadena |
| **var** | Nombre de la variable exportada de ámbito para el resultante valor de la prueba de condición. El tipo de la variable con ámbito es Booleano. | Cadena |
| **scope** | Ámbito para var (variable). | Cadena |

| **<c:choose>** | |
|---|---|
| **Descripción** | Proporciona el contexto para la ejecución condicional mutuamente excluyentes. |
| **Ejemplo de Uso** | **<c:choose>**<br>**contenido del cuerpo (<when> and <otherwise> subtags)**<br>**</c:choose>** |

***Atributos***

| Nombre | Descripción | Tipo |
|---|---|---|
| **ninguna** | | |

| <c:when> | | |
|---|---|---|
| **Descripción** | Representa una alternativa dentro de un <c: choose> acción. | |
| **Ejemplo de Uso** | **<c:when prueba="${varA < 5}"><br>contenido del cuerpo<br></c:when>** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **Test** | La condición de prueba que determina si el contenido del mensaje debe ser procesado. | Booleano |

| <c:otherwise> | | |
|---|---|---|
| **Descripción** | Representa la última alternativa dentro de un <c: choose> acción. | |
| **Ejemplo de Uso** | **<c:otherwise><br>contenido del cuerpo<br></c:otherwise>** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **ninguna** | | |

## Acciones de iterador

Las acciones del bucle sobre colecciones, para un número fijo de veces, o más de un conjunto de tokens de cadena. Estas acciones comparten los siguientes atributos para una iteración un subconjunto de elementos.

| <c:foreach> | |
|---|---|
| **Descripción** | Repite su contenido cuerpo anidado sobre una colección de objetos, o lo repite un número fijo de veces. |
| **Ejemplo de Uso** | **Iteración en una colección de objetos<br><c:forEach var="libro" items="${bookList.rows}"<br>begin='0' end='2'><br>contenido del cuerpo<br></c:forEach><br>Iteración sobre un número fijo de elementos<br><c:forEach begin="1" end="${totalSize - 1}" ><br>contenido del cuerpo<br></c:forEach>** |
| ***Atributos*** | |

| Nombre | Descripción | Tipo |
|---|---|---|
| **var** | Nombre de la variable para el elemento actual de la iteración. Esto con ámbito la variable ha anidado visibilidad. Su tipo depende en el objeto de la colección subyacente. | Cadena |
| **varStatus** | Nombre de la variable para el estado de la iteración. Objeto exportado es de tipo javax.servlet.jsp.jstl.core. LoopTagStatus. Esta variable tiene alcance visibilidad anidada. | Cadena |
| **items** | Colección de artículos para iterar encima. | J2E Collection type int |
| **begin** | Si los artículos especificados: La iteración se inicia en el punto situado en el especificado índice. En primer elemento de la colección tiene el índice 0.<br>Si los artículos no especificados: La iteración comienza con índice fijado en el valor especificado. | J2E Collection type int |
| **end** | Si los artículos especificados: Iteración termina en el punto situado en el especificado índice (ambos inclusive).<br>Si los artículos no especificados: Iteración termina cuando el índice alcanza el valor especificado. | Entero |
| **step** | Iteración únicamente tratará cada artículo paso a paso de la colección, a partir del primero. | Entero |

| **<c:forToken>** | | |
|---|---|---|
| **Descripción** | Repite fichas, separadas por los delimitadores suministrados. | |
| **Ejemplo de Uso** | **<c:forTokens var="token" items="${tokens}"**<br>**delims="," >**<br>**contenido del cuerpo**<br>**</c:forTokens>** | |
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **Var** | Nombre de la variable para el elemento actual de la iteración. Esto con ámbito la variable ha anidado visibilidad. | Cadena |
| **varStatus** | Nombre de la variable para el estado de la iteración. Objeto exportado es de Tipo javax.servlet.jsp.jstl. core.LoopTagStatus. Esta variable alcance se ha anidado visibilidad. | Cadena |
| **Ítems**<br>**delims** | Serie de fichas para repetir.<br>El conjunto de delimitadores (los caracteres que separan las señales de la serie). | Cadena<br>Cadena |
| **begin** | Iteración comienza en el token situada en el índice especificado. Primer token tiene el índice 0. | Entero |
| **end** | Iteración termina en el token situada en el índice especificado (inclusivo). | Entero |
| **step** | Iteración sólo se procesará cada paso de la señal de la cadena, a partir con la primera. | Entero |

## URL relacionados a acciones

Acciones para importar contenido de las direcciones URL, la construcción URL y redirigir.
Acciones para importar contenido de las direcciones URL, la construcción de URL y redirigir.

| **<c: import>** | |
|---|---|
| **Descripción** | Importa el contenido de un recurso basado en URL. |
| **Ejemplo de Uso** | **Recursos en línea de contenido o exportado como un objeto String**<br>**<C: url import = "/ header.jsp" var = "header">**<br>**contenido del cuerpo opcional para <c: param> subetiquetas**<br>**</ C: import>**<br>**contenido del recurso exportado como un objeto Reader**<br>**<C: url import = "/ Authors.xml"**<br>**varReader = "lector">**<br>**contenido del cuerpo donde es consumido por varReader, otra acción**<br>**</ C: import>** |

***Atributos***

| Nombre | Descripción | Tipo |
|---|---|---|
| **URL** | La dirección URL del recurso para importar. | Cadena |
| **Contexto** | Nombre del contexto cuando acceder a una dirección URL relativa recurso que pertenece a una contexto extranjero. | Cadena |
| **Var** | Nombre de la variable para con ámbito para la URL procesada. El tipo de la variable con ámbito es Cadena. | Cadena |
| **Scope** | Ámbito de la variable | Cadena |

| **<c: redirect>** | |
|---|---|
| **Descripción** | Envía una redirección HTTP al cliente. |
| **Ejemplo de Uso** | **<c:redirect url="http://www.mkp.com" />** |

***Atributos***

| Nombre | Descripción | Tipo |
|---|---|---|
| **URL** | La dirección URL del recurso para redirigir a. | Cadena |
| **Contexto** | Nombre del contexto cuando volver a dirigir a un pariente (otra) recursos URL que pertenece a un contexto exterior. | Cadena |

| **<c: param>** | |
|---|---|
| **Descripción** | Envía una redirección HTTP al cliente. |

| **Ejemplo de Uso** | **<c:import url="/header.jsp" ><br><c:param name="titulo"<br>value=" Las muestras de importación " /><br></c:import>** | |
|---|---|---|
| ***Atributos*** | | |
| **Nombre** | **Descripción** | **Tipo** |
| **Name** | Nombre de la cadena de consulta con parámetro. | Cadena |
| **Value** | Valor del parámetro. | Cadena |

## 3. FMT Tag Library

Etiquetas de Formato (Formatting tags)

Este tipo de etiquetas ayudan a dar formatos específicos a elementos dentro de un JSP. Por ejemplo: fechas, monedas, números. Ayuda al formateo basado en el "Locale/i18n" del usuario.

Si se utilizará la librería FMT podemos importarla en la página que se necesitará, para ello sólo se debe de escribir el correspondiente taglib al inicio de la página JSP:

**<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>**

### Acciones de internacionalización (I18N)

| **<fmt:setLocale>** | |
|---|---|
| **Descripción** | Se utiliza para establecer la configuración regional predeterminada para el ámbito especificado, esto anularía las locales basadas en el navegador |
| **Ejemplo de Uso** | **<fmt:setLocale scope="session" value="en_US" >** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string o java.util.Locale)** | Cadena que representa a un local. Su valor por defecto es en_US, es el único atributo que es realmente requerido para esta etiqueta. |
| **variant** | Variante de configuración regional a especificar junto con el local (idioma y país). |
| **scope (string)** | Especifica el alcance de la variable de configuración local, puede ser cualquiera de los ámbitos estándar de JSP, page, request, session o application. |

| **<fmt:bundle>** | |
|---|---|
| **Descripción** | Se usa para establecer el contexto de localización, basado en el recurso especificado, para ser usado dentro del cuerpo de esta etiqueta. |
| **Ejemplo de Uso** | **<fmt:bundle basename="Recurso" prefix="label."/>**<br>**<fmt:message key="título">**<br>**<fmt:bundle>** |
| **Atributos** | |
| **Nombre** | **Descripción** |
| **basename** | Se usa para especificar el nombre completo del recurso a utilizar para el paquete de recursos, se parece a la referencia a clases usando la notación "." |
| **prefix** | Es el valor a anteponer al atributo clave de cualquier anidado <fmt:message> |

| <fmt:setBundle> | |
|---|---|
| **Descripción** | Proporciona una forma de crear un contexto de internacionalización-localización y luego proporciona una manera de almacenar una variable con ámbito definido |
| **Ejemplo de Uso** | **<fmt:setBundle basename="Mensajes" var="lang"/>** |
| ***Atributos*** | |
| **Nombre** | **Descripción** |
| **basename** | Nombre del recurso para exponer como una variable con ámbito o configuración |
| **var** | Nombre de la variable que almacena el contexto de localización, su valor por defecto es I10n |
| **scope** | Ámbito de la variable var, su valor por defecto es: page. |

| <fmt:message> | |
|---|---|
| **Descripción** | Son los mensajes en cadena que parecen en las páginas JSP, se puede utilizar sin cualquier contenido del cuerpo o puede contener la etiqueta <fmt:param> que sirve para la sustitución del formato del mensaje. El mensaje final se imprime o se almacena en una variable |
| **Ejemplo de Uso** | **<fmt:message key="count.ten" bundle="${lang}" />** |
| ***Atributos*** | |
| **Nombre** | **Descripción** |
| **key** | Tecla de mensajes a recuperar, su valor por defecto es body |
| **scope** | Ámbito de la variable para almacenar el mensaje localizado |
| **var** | Nombre de la variable para almacenar el mensaje localizado |
| **bundle** | Contiene un contexto de localización específica a utilizar para la búsqueda clave del mensaje |

| <fmt:param> | |
|---|---|
| **Descripción** | Suministra un parámetro de sustitución de formato del mensaje que contiene <fmt:message>. Los parámetros se sustituyen en orden secuencial. |
| **Ejemplo de Uso** | **<fmt:param value = "<string>">** |
| ***Atributos*** | |
| **Nombre** | **Descripción** |
| **value** | Valor que se usa para el formato de mensaje de sustitución paramétrica. |

| <fmt:requestEnconding> | |
|---|---|
| **Descripción** | Se usa para especificar el tipo de codificación de los caracteres de la solicitud y decodifica los formularios de entrada de datos introducidos por el usuario. |
| **Ejemplo de Uso** | **<fmt:requestEnconding value="UTF-8" />** |
| ***Atributos*** | |
| **Nombre** | **Descripción** |
| **value** | Codificación de caracteres a utilizar. |

## Acciones de Formato

Se usan para los datos que contienen fechas, números y horas.

| <fmt:timeZone> | |
|---|---|
| **Descripción** | Establece la zona horaria para ser aplicada al contenido del cuerpo anidado. |
| **Ejemplo de Uso** | **<fmt:timeZone value=”GMT+07”> Hora actual: <fmt:formatDate type=”time” timeStyle=” short” value=”${now}”> </fmt:timeZone>** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string o java.util.TimeZone)** | Indica la zona horaria, si es una cadena se interpreta como un ID de zona horaria. Si no se especifica se utiliza la zona GMT (hora del meridiano de Greenwich). |

| <fmt:setTimeZone> | |
|---|---|
| **Descripción** | Establece la zona horaria especificada en una variable de ámbito con nombre o utilizando el nombre de la zona horaria por defecto si no se especifica el atributo var. |
| **Ejemplo de Uso** | **<fmt:setTimeZone value=”${param.currentTimeZone}” scope=”application”>** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string o java.util.TimeZone)** | Especifica la zona horaria |
| **var (string)** | Nombre de la variable que contendrá la zona horaria |
| **scope (string)** | Ámbito de la variable de configuración horaria |

| <fmt:formatNumber> | |
|---|---|
| **Descripción** | Aplica formato a un número, moneda o porcentaje de una manera sensible a la localidad. Puede tener contenido en el cuerpo o no. |
| **Ejemplo de Uso** | **<fmt:formatNumber value="12" type="number"/ >** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string o number)** | Valor numérico a formatear |
| **type (string)** | Indica el tipo de formato a utilizar, puede ser: number, currency, percent |
| **pattern (string)** | Patrón de formato particularizado |
| **currencyCode (string)** | Código ISO 4217 (ejemplo: CAD: Dólar canadiense, USD: dólar estadounidense, EUR: euro, etc.) |
| **currencySymbol (string)** | Símbolo de moneda |
| **groupingUsed (boolean)** | Indica si la salida tendrá separador de miles |
| **maxIntegerDigits (int)** | Máximo de dígitos en la parte entera de la salida |
| **minIntegerDigits (int)** | Mínimo de dígitos en la parte entera de la salida |
| **maxFractionDigits (int)** | Máximo de dígitos en la parte decimal de la salida |
| **minFractionDigits (int)** | Mínimo de dígitos en la parte decimal de la salida |
| **var (string)** | Nombre de la variable para almacenar la salida como un string |
| **scope (string)** | Ámbito de var, no se puede especificar si no se especifica var |

| <fmt:parseNumber> | |
|---|---|
| **Descripción** | Se usa para analizar una representación en forma de cadena de un número, moneda o porcentaje en formato sensible al lenguaje. Puede tener contenido en el cuerpo o no. |
| **Ejemplo de Uso** | **<fmt:parseNumber var="fmtdNum" type="number" value="145.43" integerOnly="true" >** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string)** | Cadena de caracteres a analizar |
| **type (string)** | Indica se value se analizará como número, moneda o como porcentaje |
| **pattern (string)** | Formato particularizado que determina como la cadena del atributo value tiene que ser analizada |
| **parseLocale (string o java.utilLocale)** | Representación de una cadena de un local utilizado para el análisis sintáctico |
| **integerOnly (boolean)** | Indica si sólo se analizará la parte entera del valor especificado |
| **var (string)** | Nombre de la variable que guardará el resultado |
| **scope (string)** | Ámbito del atributo var |

| <fmt:formatDate> | |
|---|---|
| **Descripción** | Analiza una cadena que representa una fecha Y/U hora de una manera sensible a la localidad. |
| **Ejemplo de Uso** | **<fmt:formatDate pattern=”yy-mm-dd” value=”${now}”>** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string)** | Fecha y/o hora a aplicar formato |
| **type (string)** | Indica si la hora, la fecha o ambos serán formateados (tipos definidos: time, date, both) |
| **dateStyle (string)** | Estilo de formato de la fecha (default, short, médium, long, full) |
| **timeStyle (string)** | Estilo de formato de la hora (default, short, médium, long, full) |
| **pattern (string)** | Formato particularizado que determina como debe ser analizado el atributo value |
| **timeZone (string o java.util.TimeZone)** | Zona horaria |
| **parseLocale (string o java.utilLocale)** | Localización en la cual se utilizará el patrón de formato por defecto cuando se realice la operación del análisis. |

| <fmt:parseDate> | |
|---|---|
| **Descripción** | Analiza la representación en forma de cadena de caracteres de una fecha y hora en un formato sensible al lenguaje. Puede tener contenido en el cuerpo o no. |
| **Ejemplo de Uso** | **<fmt:parseDate var=”date01” type=”date” dateStyle=”short” value=”${now}”>** |
| ***Atributos*** | |
| **Nombre (tipo)** | **Descripción** |
| **value (string)** | Fecha y/o hora a aplicar formato |
| **type (string)** | Indica si value sólo contiene una hora o una fecha o ambos (time, date, both) |
| **dateStyle (string)** | Estilo de formato de la fecha (default, short, médium, long, full) |
| **timeStyle (string)** | Estilo de formato de la hora (default, short, médium, long, full) |
| **pattern (string)** | Formato particularizado que determina como debe ser analizado el atributo value |
| **timeZone (string o java.util.TimeZone)** | Zona horaria |
| **parseLocale (string o java.utilLocale)** | Localización en la cual se utilizará el patrón de formato por defecto cuando se realice la operación del análisis. |
| **var (string)** | Nombre de la variable que almacenará el resultado |
| **scope (string)** | Ámbito del atributo var |

## 4. SQL Tag Library

La librería de etiquetas SQL proporciona etiquetas para interactuar con bases de datos relacionales, tales como ORACLE, My SQL y Microsoft SQL server.

Para incluir esta librería de etiquetas en la página JSP se debe agregar la siguiente sintaxis:

**<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>**

### Acciones Generales de SQL Tag Library

| **<sql:setDataSource>** | |
|---|---|
| **Descripción** | Establece la variable de configuración de fuente de datos o guarda la información de fuente de datos en una variable con ámbito que se puede utilizar como entrada para las otras acciones de base de datos JSTL. |
| **Ejemplo de Uso** | **<sql:setDataSource var="datasource"**<br>**driver="org.gjt.mm.mysql.driver"**<br>**url="jdbc:mysql://localhost/db"**<br>**user="guest"**<br>**password="guest"/>**<br>**<sql:query datasource="${datasource}" … />** |
| ***Atributos*** | |
| **Nombre** | **Descripción** |
| **driver** | Nombre de la clase controlador JDBC a ser registrada. |
| **url** | Dirección JDBC para la conexión de base de datos. |
| **user** | Nombre de usuario en la base de datos. |
| **password** | Contraseña de base de datos. |
| **dataSource** | Base de datos preparada de antemano. |
| **var** | Nombre de la variable que representa la base de datos. |
| **scope** | Ámbito de la variable que representa la base de datos. |

| **<sql:query>** | |
|---|---|
| **Descripción** | Ejecuta un SQL SELECT statement y guarda el resultado en una variable con ámbito. |
| **Ejemplo de Uso** | **SQL definido en el atributo**<br>**<sql:query sql="SELECT * FROM books WHERE title = 'JSTL' ORDER BY author"**<br>**var="titles" dataSource="${datasource}" >**<br>**</sql:query>**<br><br>**SQL en contenido del body**<br>**<sql:query var="titles" dataSource="${datasource}" >**<br>**SELECT * FROM books WHERE title = 'JSTL' ORDER BY author**<br>**</sql:query>**<br><br>**Suministrando un parámetro en la consulta.**<br>**<sql:query sql="SELECT * FROM books WHERE title = ? ORDER BY author"**<br>**var="titles" dataSource="${datasource}" >** |

```
        <sql:param value="${titleSelected}"/>
</sql:query>

Consulta usando maxRows y startRow
<sql:query sql="SELECT * FROM books WHERE title = 'JSTL' ORDER BY
          author" var="titles" dataSource="${datasource}"
          maxRows="5"
          startRow="7" >
</sql:query>
```

***Atributos***

| Nombre | Descripción |
|---|---|
| **sql** | Comando SQL a ejecutar (debe devolver un Resultset). |
| **dataSource** | Conexión de base de datos a utilizar. |
| **maxRows** | Número máximo de resultados para almacenar en la variable. |
| **startRow** | Numero de fila en el resultado, para iniciar la inserción de datos. |
| **var** | Nombre de la variable que representa a la base de datos. |
| **scope** | Ámbito de aplicación de la variable para exponer el resultado de la base de datos. |

| **<sql:update>** | |
|---|---|
| **Descripción** | Ejecuta un SQL statement que no retorna datos, por ejemplo SQL INSERT, UPDATE, o DELETE statements. |
| **Ejemplo de Uso** | `<sql:update dataSource="$(datasource)" var="updateResult"`<br>`scope="request">`<br>`UPDATE book SET Title = 'JSTL First Edition' WHERE`<br>`author = 'Spielman'`<br>`</sql:update>` |

***Atributos***

| Nombre | Descripción |
|---|---|
| **sql** | Comando SQL a ejecutar (no debe devolver un Resultset). |
| **dataSource** | Conexión a usar para la base de datos (Anula el valor predeterminado). |
| **var** | Nombre de la variable para almacenar el conteo de las filas modificadas. |
| **scope** | Ámbito de la variable para almacenar el recuento de filas modificadas. |

| <sql:param> | |
|---|---|
| **Descripción** | Usado como una acción anidada para <sql:query> y <sql:update> para proporcionar un valor para un marcador de posición de valor. Si se proporciona un valor nulo, el valor se establece SQL NULL para el marcador de posición. |
| **Ejemplo de Uso** | **<c:set var="firstName">Bill</c:set><br><sql:query var="users"><br>SELECT user_name, last_name<br>FROM users WHERE first_name = ?<br><sql:param value="${firstName}"/><br></sql:query>** |

***Atributos***

| Nombre | Descripción |
|---|---|
| **value** | Valor del parámetro a modificar. |

| <sql:dateParam> | |
|---|---|
| **Descripción** | Se utiliza como una acción anidada para <sql:query> y <sql:update>, suministra un valor de fecha y hora de un marcador de posición de valor. Si se proporciona un valor nulo, el valor se establece SQL NULL para el marcador de posición. |
| **Ejemplo de Uso** | **<fmt:parseDate var="ww2End"<br>pattern="yyyy-MM-dd"<br>value="1945-09-02"/><br><sql:query var="postWarBabies"><br>SELECT user_name FROM user_profile<br>WHERE birth_date > ?<br><sql:dateParam value="${ww2End}"<br>type="date"/><br></sql:query>** |

***Atributos***

| Nombre | Descripción |
|---|---|
| **value** | Valor del parámetro para establecer la fecha (java.util.Date). |
| **type** | DATE (solo fecha), TIME (solo hora), o TIMESTAMP (fecha y hora). |

| <sql:transaction> | |
|---|---|
| **Descripción** | Se utiliza para agrupar <sql:query> y <sql:update> en las transacciones. Se puede poner el mayor número <sql:query> y <sql:update>, como sentencias dentro de <sql:transaction> para que sean una sola transacción. Se asegura de que las modificaciones de base de datos realizadas por las acciones anidadas sean cometidas o bien confirma se deshace, si se |

| | |
|---|---|
| | produce una excepción por una acción anidada. |
| **Ejemplo de Uso** | **<sql:transaction isolation="serializable"><br><sql:update><br>UPDATE books SET edition="2" WHERE author="Spielman"<br></sql:update><br><sql:query var="bookList" ><br>SELECT * FROM books ORDER BY author<br></sql:query><br></sql:transaction>** |

***Atributos***

| Nombre | Descripción |
|---|---|
| **dataSource** | Conexión a la base de datos a utilizar. |
| **isolation** | Aislamiento de la transacción (READ_COMMITTED, READ_UNCOMMITTED, REPEATABLE_READ, SERIALIZABLE). |

# 5. XML Tag Library

## Implementación y funciones útiles

La librería XML soporta la conversión de documentos XML, selección de fragmentos XML, procesamiento basado en condición e iteración de contenido XML y transformaciones XSLT. Un patrón común para el uso de etiquetas XML es como los siguientes:

- Uso de <x:parse> para convertir XML en una variable del ámbito. Puede usarse literalmente del cuerpo (body) utilizando la etiqueta <c:import> o del atributo de valor que debe referirse a cualquier fuente XML: <x:parse var="nombreVariable"…>

- Utilizar la variable del ámbito de <x:parse> para especificar el documento XML utilizado en expresiones XPath. Por ejemplo: <x:out select="$nombreVar/XPath expressions" />

Para utilizar la librería se debe declarar en el encabezado de la página de la siguiente forma:

**<%@ taglib prefix="x" uri="http://java.sun.com/jstl/xml" %>**

De esta forma podremos utilizar las etiquetas "x" para llamar a cualquier librería del conjunto XML.

Se pueden utilizar datos en el mismo ámbito de la página web como referencia para llamar a una expresión en formato XML utilizando variables XPath (similar uso a las variables en EL). Para esto se crea una estructura XML con la etiqueta <x:parse>. Por ejemplo:

```
<x:parse var="Librería">
        <libros>
                <libro id="200">
                        <nombre>Tom Sawyer</nombre>
                        <autor>Mark Twain</autor>
                </libro>
                <libro id="201">
                        <nombre>El Principito</nombre>
                        <autor>Antoine de Saint-Exupéry</autor>
                </libro>
                <libro id="202">
                        <nombre>Mil años de soledad</nombre>
                        <autor>Gabriel García Marquez</autor>
                </libro>
        </libros>
</x:parse>
<%--Definiendo la variable que contendrá los datos --%>
<c:set var="idLibro" value="${libro.id}" scope="page" />
<x:set var="libro" select="$Librería//libros/libro[@id=$pageScope:idLibro]" />
```

Está viendo el libro: **<x:out select="$libro/nombre" />**

Detalles de sintaxis abreviada:

| Abreviatura | XPath | Ejemplo |
|---|---|---|
| **//** | Decendente | $Librería//nombre |
| **.** | A si mismo | $Librería/libros/libro/. |
| **..** | Padre | $Librería//[nombre='Tom Sawyer']/../nombre |
| **@** | Atributo | $Librería//libro[@id="201"]/autor |
| **(espacio)** | Hijo | $Librería/libros/libro[nombre='El Principito'] |
| * | Coincidencia cualquier caracter | $Librería//*[local-name()='libros']/libro |

También existen funciones que pueden ser usadas en expresiones XPath.

| Nombre | Argumentos | Retorno | Descripción |
|---|---|---|---|
| **ceiling** | Número u objeto | Número | Retorna el entero más próximo mayor o igual a un número enviado como argumento. |
| **concat** | String1, String2,…, n | String | Concatena una cadena de caracteres hasta "n" |
| **contains** | String1, String2 | Boolean | Compara si la segunda cadena contiene caracteres de la primer cadena |
| **count** | Set de nodos | Número | Devuelve la cantidad de nodos que se encuentran en un conjunto |
| **floor** | Número u objeto | Número | Retorna el entero más próximo menor o igual al número enviado como argumento. |
| **last** | | Número | El número de nodos en el contexto |
| **not** | Boolean | Boolean | Retorna la inversión lógica del argumento brindado, si argumento = true, retorna false y viceversa. |
| **round** | Número | Número | Retorna el entero más cercano al número enviado redondeando a partir de 0.5 hacia el mayor y debajo de 0.5 al menor. |
| **starts-with** | String1, String2 | Boolean | Retorna verdadero si la cadena 2 inicia con la cadena 1 |
| **string-length** | Objeto | Número | Retorna la cantidad de caracteres de una cadena. Si el objeto no es una cadena, lo convierte a string y luego cuenta los caracteres. |
| **substring** | String1, index, length | String | Retorna una sub cadena de una cadena iniciando en "index" y continuando durante "length" caracteres. El primer carácter está en la posición 1 no en la posición 0. |

También se pueden importar objetos XML de una URL específica, por ejemplo para procesar XML de tipo Really Simple Syndication (RSS) para sindicar o compartir contenido en la web.

Un archivo de este tipo se puede encontrar con la siguiente estructura:

```
<?xml versión=”1.0” ?>
<rss versión=”0.91”>
        <channel>
                <title>Título del anuncio</title>
                <link>URL fuente</link>
                <description>Descripción del anuncio</description>
                <ítem>
                        <title>Título del artículo 1</title>
                        <link>URL del artículo 1</link>
                        <description>Descripción del artículo 1</description>
                </ítem>
                <ítem>
                        <title>Título del artículo 2</title>
                        <link>URL del artículo 2</link>
                        <description>Descripción del artículo 2</description>
                </ítem>
        </channel>
</rss>
```

## Acciones generales de XML

| **< x:parse>** | |
|---|---|
| **Descripción** | Convierte el contenido XML brindado por el valor del atributo o las etiquetas del “body” en un ámbito de variables. Estas variables pueden ser usadas para procesamiento de otras etiquetas XML. |
| **Ejemplo de Uso** | **<c:import var=”rss” url=”http://servlet.java.sun.com/syndication/rss_java_highlights-PARTNER-20.xml” /><br><x:parse var=”noticias” xml=”${rss}” />** |

***Atributos***

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|
| **doc** | Texto del documento para una conversión como cadena o lectura | No | Body |
| **systemId** | URI del documento original usado para resolución de entidad. | No | Body |
| **filter** | Filtro XML a aplicar. El valor debe ser de tipo org.xml.sax.XMLFilter | No | Ninguno |
| **var** | Nombre de la variable a almacenar el resultado | Si | Ninguno |
| **scope** | Define el ámbito de la variable “var” | No | Page |
| **varDom** | Nombre de la variable a almacenar resultado (guardada como un objeto DOM) | Si | Ninguno |
| **scopeDom** | Ámbito de almacenamiento de la variable varDom | No | Page |

| < x:out> | | | |
|---|---|---|---|
| **Descripción** | Imprime el resultado de XPath como una cadena. | | |
| **Ejemplo de Uso** | <x:out select=”$noticias//channel/title” /> | | |
| ***Atributos*** | | | |
| **Nombre** | **Descripción** | **Requerido** | **Por defecto** |
| **select** | Expresión XPath | Si | Ninguno |
| **escapeXml** | True para imprimir caracteres especiales. | No | True |

| < x:set> | | | |
|---|---|---|---|
| **Descripción** | Guarda el resultado de una expresión XPath select en una variable del ámbito. El valor de retorno está en un nodo (fragmento XML) y son de tipo boolean, string o número. | | |
| **Ejemplo de Uso** | **<x:set select=”$Noticias//channel/item” var=”newsItems” />** | | |
| ***Atributos*** | | | |
| **Nombre** | **Descripción** | **Requerido** | **Por defecto** |
| **select** | Expresion XPath | No | Ninguno |
| **var** | Nombre de la variable para almacenar resultados, el tipo de variable es igual a cualquier valor que es retornado de la expresión select. Boolean: java.lang.Boolean, Number: java.lang.Number y Node o Node set: implementación del tipo dependiente. | No | Ninguno |
| **scope** | Ámbito de la variable “var” | No | Page |

## Acciones de control de flujo

También se pueden representar acciones de control de flujo como se vio en las etiquetas “core”.

| < x:if> | |
|---|---|
| **Descripción** | Procesa el contenido si la selección de XPath retorna verdadero. |
| **Ejemplo de Uso** | **<x:if select=”$Noticias//ítem/[contains(description, ‘Linux’)]”><br>Las noticias linux están aquí<br></x:if>** |
| ***Atributos*** | |

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|
| **select** | Condición de prueba de una expresión XPath. El contenido es desplegado si la condición es verdadera | Si | Ninguno |
| **var** | Nombre de la variable para almacenar el resultado de la condición | No | Ninguno |
| **scope** | Ámbito de la variable "var" | No | Page |

| **< x:choose >** | |
|---|---|
| **Descripción** | De igual forma que la "core" tag library, este objeto es utilizado para estas condiciones en conjunto con las etiquetas <x:when> y <x:otherwise>. Tiene forma de un switch o "Select case" en la programación cotidiana y se auxilia de las demás etiquetas mencionadas como un conjunto. |
| **Ejemplo de Uso** | **<x:choose><br>  <x:when select="$Noticias//ítem"><br>  Tenemos nuevas noticias<br>  </x:when><br>  <x:otherwise><br>  ¡NO hay noticias hoy!<br>  </x:otherwise><br></x:choose>** |

***Atributos***

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|
| **ninguno** | | | |

| **< x:forEach >** | |
|---|---|
| **Descripción** | Esta instrucción es de mucha utilidad para iterar entre todos los nodos que forman un conjunto estableciendo la dirección XPath en la selección. |
| **Ejemplo de Uso** | **<x:forEach select="$Noticias//ítem"><br>  <a href='<x:out select="link" />'><br>  <x:out select="title" /></a><br /><br></x:forEach>** |

***Atributos***

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|
| **select** | Nombre de la variable a establecer en el objeto actual. Variable solamente puede ser usada en anidamiento de esta etiqueta | No | Ninguno |
| **var** | Nombre de la variable que contendrá el objeto. | No | Ninguno |
| **scope** | Ámbito de la variable "var" | No | Page |

## Acciones de transformación

| < x:transform > | |
|---|---|
| **Descripción** | Proporciona una transformación XSLT brindada por el atributo “doc” o el cuerpo de la etiqueta. La hoja de estilo XSL es especificada por el atributo xslt. Cuando en la mayoría de casos será establecida programáticamente. |
| **Ejemplo de Uso** | |

***Atributos***

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|
| **doc** | Fuente XML del documento para trasformación. El valor puede ser una cadena, lectura, javax.xml.transform.Source, org.w3c.dom.Document o cualquier valor exportado como <x:parse> o <x:set> (si el valor viene de un <x:set>, no puede ser un documento parcial). | No | Body |
| **docSystemId** | URI de la fuente XML usada para resolver referencias de entidad. | No | Ninguno |
| **xslt** | Hoja de estilo XSLT para usar. El valor debe ser una cadena, Reader o un objeto javax.xml.transform.Source | Si | Ninguno |
| **var** | Variable para obtener el resultado como un objeto de tipo org.w3c.doc.Document | No | Ninguno |
| **scope** | Ámbito de la variable “var” | No | Page |
| **result** | Objeto javax.xml.transform.Result que contiene o procesa el XML transformado. | No | Ninguno |

| < x:param > | |
|---|---|
| **Descripción** | Establece el parámetro de transformación que será pasado a la hoja de estilo que se declara como la etiqueta <xsl:param>. La etiqueta <x:param> solo puede ser anidada dentro de la etiqueta <x:transform>. Cualquier etiqueta <x:param> debe venir después del contenido de body XML de la etiqueta <x:transform> |
| **Ejemplo de Uso** | **<x:transform xml=”${Noticias}” xslt=”${searchXsl}” /><br><x:param name=”searchParam”><br>Servicios Web<br></x:param><br></x:transform>** |

***Atributos***

| Nombre | Descripción | Requerido | Por defecto |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **name** | Nombre del parámetro como cadena. Este debe coincidir con el nombre correspondiente en la etiqueta <xsl:param> de XSTL | Si | Ninguno |
| **value** | Valor del parámetro. Si no está especificado el parámetro, se tomará de la etiqueta del cuerpo | No | Body |

# Conclusiones

Basado en el contenido del documento presente, se puede sintetizar lo siguiente:

- Se ha logrado brindar una base introductoria del material de JSTL y EL permitiendo al lector hacer uso de estas etiquetas cuando considere necesario.

- Se ha demostrado la facilidad con que las etiquetas son utilizadas brindando ejemplos eficientes y prácticos sobre su uso.

Se ha determinado que por medio del manual se puede guiar al lector a profundizar el uso de las etiquetas JSTL y EL ampliando sus conocimientos en el ámbito técnico de esta materia.

# Bibliografía


Tutorial de etiquetas JSTL
http://docs.oracle.com/javaee/5/tutorial/doc/bnakc.html

Tutorial de JSTL
https://www.youtube.com/watch?v=68-2C9L5dJM

Tutorial y uso de JSTL
http://desarrollo-java.readthedocs.org/es/latest/tutorial6.md.html

Tutorial introductorio de JSTL
http://www.javamexico.org/blogs/kaztle_8/jstl_core_jsp_parte_4

Manual de JSTL
https://lookaside.fbsbx.com/file/manual%20de%20JSTL.pdf?token=AWy2-N2jy-G-9bgCINgWcott9raGckr1zmGoFJVwrHVQb3XJ5RrltlTP5nAiobWVDe-HtTo8WZRnk4hk-kwrCZhqZ0J2AqRcA8oTwiFOjuu3evm2RzaYp2UqMrHJyHSFxKWTOYPyi7gMDROMl3GeZf1J

Libro práctico de Guía de JSTL y EL
https://lookaside.fbsbx.com/file/JSP%20Practical%20Book.pdf?token=AWxEIM-a1Zdo27DkOzOwUHio4xkG7QnN5eu-_GVF8etvA5ysjB2rBUdZb1E_Pm9ookultcgDxq4RqXIfs0XB3gJ6btCaLsKXTd3FGcKpwDEMRc6Ky7zZRTuvpwnjZl891OvmmNxsiQ-qf9S-TRI5uGUL