CURSO DE SERVLETS Y JSPS

JSP STANDARD TAG LIBRARY (JSTL)

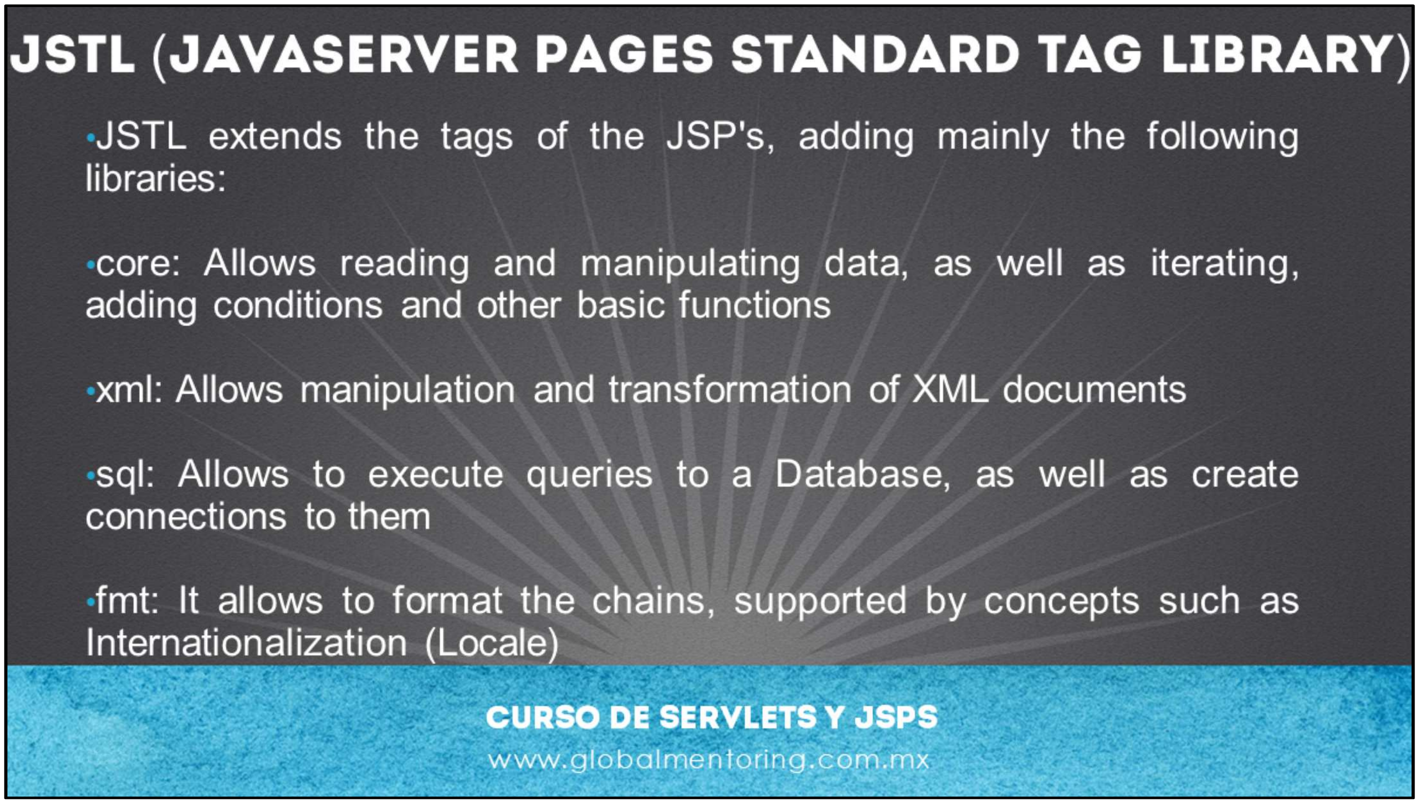
By the expert: Eng. Ubaldo Acosta

Eng. Ubaldo Acosta
Global Mentoring
JAVA
UNIVERSITY
CURSO DE SERVLETS Y JSPS
www.globalmentoring.com.mx

Hello, Ubaldo Acosta greets you again. I hope you're ready to start with this lesson.

We are going to study the theme of JSP Standard Tag Library, also known as JSTL.

Are you ready? Come on!



JSTL (JAVASERVER PAGES STANDARD TAG LIBRARY)

- JSTL extends the tags of the JSP's, adding mainly the following libraries:
- core: Allows reading and manipulating data, as well as iterating, adding conditions and other basic functions
- xml: Allows manipulation and transformation of XML documents
- sql: Allows to execute queries to a Database, as well as create connections to them
- fmt: It allows to format the chains, supported by concepts such as Internationalization (Locale)

CURSO DE SERVLETS Y JSPS
www.globalmentoring.com.mx

In this lesson we are going to review the concept of JSTL.

JSTL stands for JavaServer Pages Standard Tag Library, this is a library of the JSPs that extends the basic functionality of them, adding mainly the following libraries: core, xml, sql and fmt.

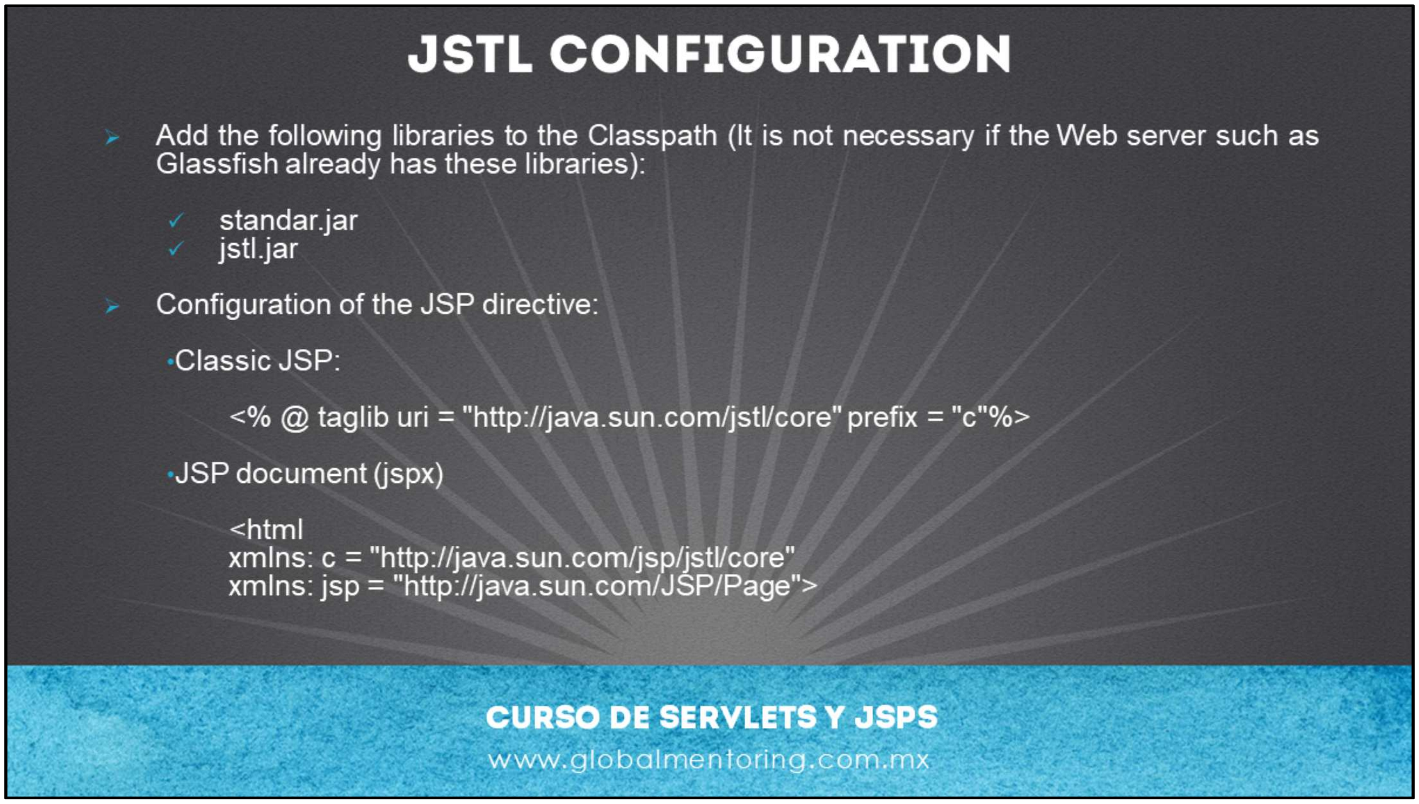
The core library will allow us to read and manipulate data as well as iterate and add conditions and other basic functionalities.

The xml library allows the manipulation and transformation of xml documents.

The sql library has tags that will allow us to execute queries to a database as well as create connections to them.

Finally the fmt library will allow us to format the chains that we are manipulating to convert them, for example, from string to number or date. Even this library allows us to support the concepts such as internationalization or local English.

In this lesson we will study some of these tags to have more knowledge of how we can extend the basic functionality of JSPs using JSTL.



JSTL CONFIGURATION

- Add the following libraries to the Classpath (It is not necessary if the Web server such as Glassfish already has these libraries):
  - ✓ standar.jar
  - ✓ jstl.jar
- Configuration of the JSP directive:
  - Classic JSP:

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

  - JSP document (jspx)

```
<html
xmlns: c = "http://java.sun.com/jsp/jstl/core"
xmlns: jsp = "http://java.sun.com/JSP/Page">
```

CURSO DE SERVLETS Y JSPS
www.globalmentoring.com.mx

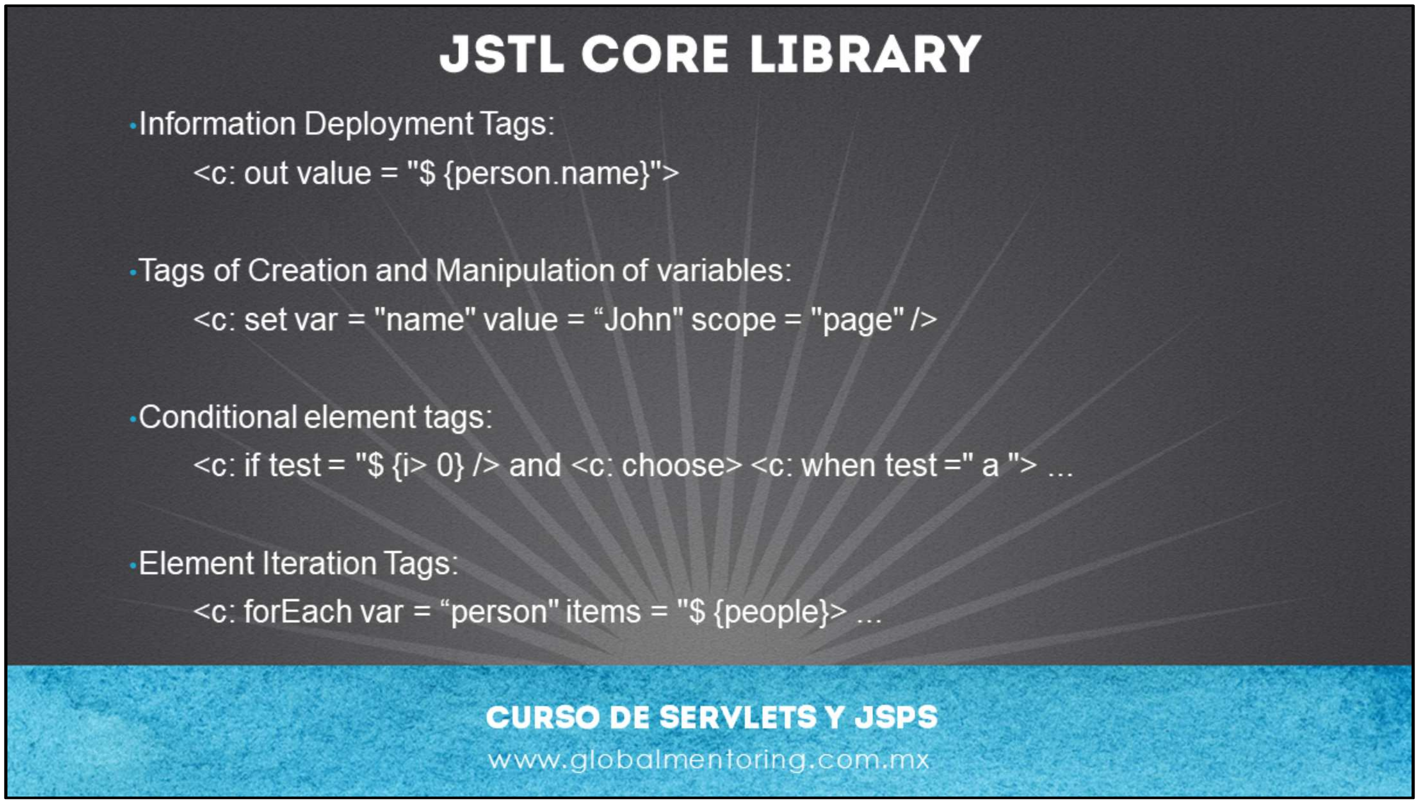
To be able to use the JSTL libraries in our web application in case we use a server like Glassfish already has the JSTL libraries, if for some reason they were not in the server we should add the .jar of standard.jar and jstl.jar to the classpath , however with Glassfish this step is not necessary.

Later to be able to use the libraries within a JSP we must add the following directive: <% @ taglib uri = "http://java.sun.com/jstl/core" prefix = "c"%>

As we can see we are using the syntax of the directives in the JSPs as we saw earlier. In this case the attribute uri = "http://java.sun.com/jstl/core" is a uri that is resolved locally on our web server by adding the jars (standard.jar, jstl.jar), by what to request from a JSP a taglib, what the web server does is to search within these jars any file with ending .tld (tag library definition), so this uri is not something that has to be searched or resolved in internet, but simply going to look for an identical uri within the jars that we have in our web application.

If we use jspx files, ie a JSP document, what we have to do is the following within our root element, for example in this case inside the html tag what we do is specify the namespace of our tag library with the core prefix and then the uri which is exactly the same as we previously specified <% @ taglib uri = "http://java.sun.com/jstl/core" prefix = "c"%> and likewise the prefix is the same as we are using for our namespace. We must remember that our namespace is only a prefix in the same way as we are specifying in the classic notation "prefix =" c ".

There are more details regarding the configuration of JSTL but in a basic way and to start using this technology with this is enough.



JSTL CORE LIBRARY

- Information Deployment Tags:
  <c: out value = "$ {person.name}">
- Tags of Creation and Manipulation of variables:
  <c: set var = "name" value = “John" scope = "page" />
- Conditional element tags:
  <c: if test = "$ {i> 0} /> and <c: choose> <c: when test =" a "> ...
- Element Iteration Tags:
  <c: forEach var = “person" items = "$ {people}> ...

CURSO DE SERVLETS Y JSPS
www.globalmentoring.com.mx

We are going to study the JSTL core library. Once we have put into practice the core library, it is easier to use the other libraries that we mentioned earlier.

Inside the jstl core library we have several tags that will allow us to display information, let's see how we can do this:

We have a prefix called c for the core tag, as specified in the JSP. Later we specify the name of the tag that we are going to use, in this case it is as if we used a function but because we are using JSPs instead of a function it is known as a tag or label and later we use certain attributes and values to finish configure these tags, for example: <c: out value = "$ {person.name}">.

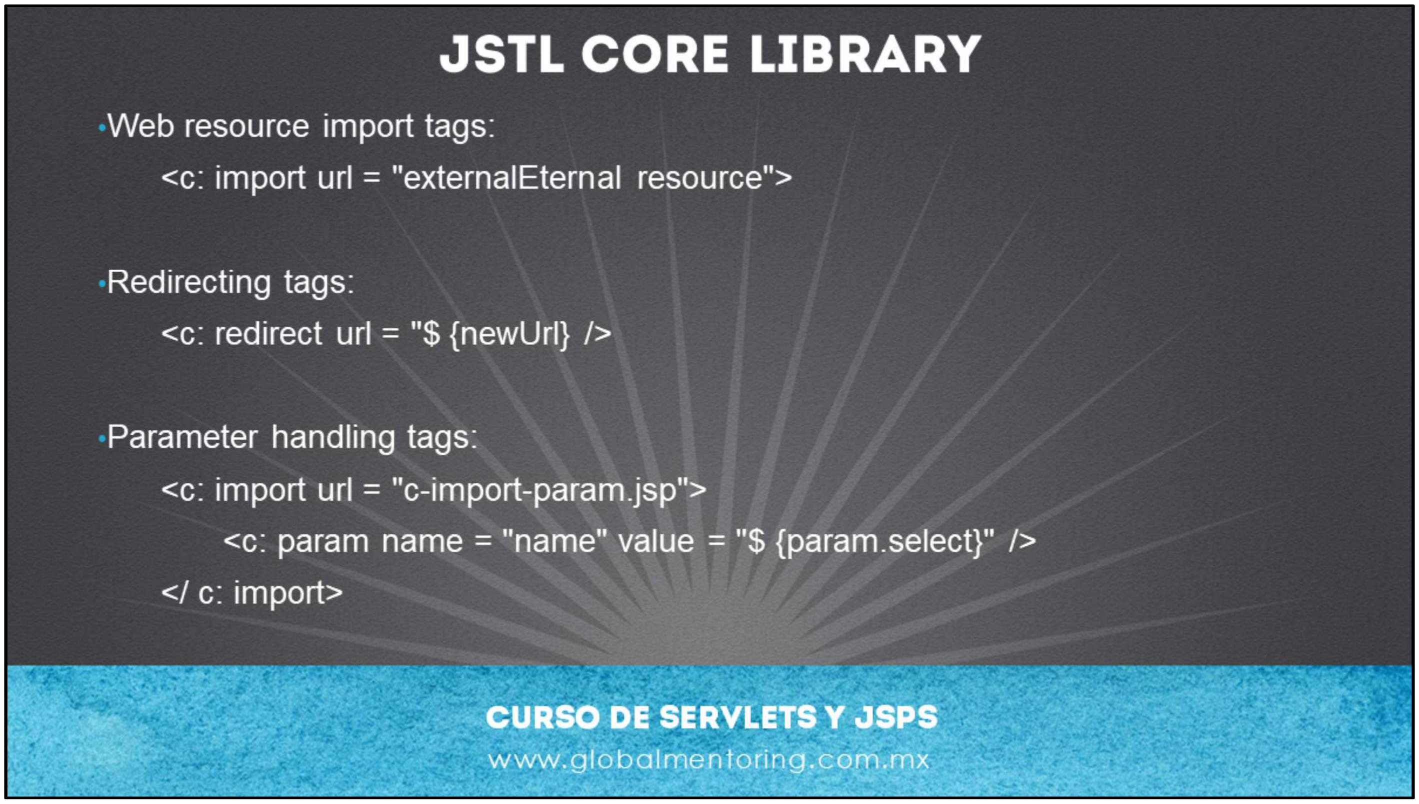
Remember that the objective of a JSP is only to manipulate tags or labels and not to manipulate Java code (although it can perform this task) so the combination of Expression Language with JSTL is very practical and are the best practices that we are going to be talking in the following exercises.

To display the value of a JavaBean, for example the Person class, and we want to access the name property, simply use the out tag and specify the value that we want to display to our client, eg. "$ {Person.name}". We could put here a static variable or a variable in hard code, but in this case we can see that by combining JSTL with Expression Language we can access the JavaBean that are in some scope that can be page, request, session or application as we have commented previously .

We also have tags to create and manipulate variables, because the Expression Language has that limitation regarding modifying JavaBeans values or creating JavaBeans. JSTL will allow us to create that type of variables with the set tag. We can declare a new variable and specify the value of said variable with <c: set var = "name" value ... and once we have created the variable we can optionally declare the scope of this variable, in this case we are specifying the scope in a way explicit scope = "page" but if we omitted this attribute scope = "page" /> it would happen that this variable name is automatically added to the page scope. So with this tag we are going to cover the limitation of expression Language to create variables and that is also why we are going to be able to omit the use of script, using code much easier to maintain and understand in a JSP.

Later we also have tags to handle conditional elements, for example the if tag if we need to handle some conditional code, eg. <c: if test = "$ {i> 0}> or also the choose tag, this is a tag very similar to the java switch, and the switch cases can be set with <c: when test =" a " >

We also have iteration tags <c: forEach var = "person" items = "{people}> ... to be able to process the elements of a list. Each of these elements will be iterated and stored in the person object. This is going to be a new complement to the Expression Language because in an Expression Language we can not iterate the elements but if we can access each of them individually, then we will combine JSTL with Expression Language to be able to access the elements of a collection.



JSTL CORE LIBRARY

- Web resource import tags:

```
<c: import url = "externalEternal resource">
```

- Redirecting tags:

```
<c: redirect url = "$ {newUrl} />
```

- Parameter handling tags:

```
<c: import url = "c-import-param.jsp">
    <c: param name = "name" value = "$ {param.select}" />
</ c: import>
```

CURSO DE SERVLETS Y JSPS

www.globalmentoring.com.mx

There are many more tags within the JSTL core but we will mention a few more.

One of the most useful tags with respect to JSTL is the tag of import <c: import url = "externalEternal resource"> in this case the difference with an include that we have previously commented with the JSPs is that the includes are only restricted to elements that are within our web application, this import tag will allow us to include internal or external resources to our web application. So we can include forms of other web applications or even complete pages of other applications. This will allow us to manipulate our application in such a way that the user does not have to change the application to be browsing in different web applications.

We also have the redirect tag <c: redirect url = "$ {newUrl}" /> this case is somewhat similar to the send redirect we have used in Servlets, but in this case it is a tag that will allow us to make a manipulation simplest of the redirect. Only what we are going to specify is the new URL, which can also be in the same way within our application or in an application external to ours.

And finally we have the parameter management tag, because we can include resources in a dynamic way we might need to send certain information to these external resources and because we can not provide parameters dynamically in this url, we will be able to combine the management of the import tag and send the parameter information by means of the param tag, eg. <c: param name = "name" value = "$ {param.select}" /> specifying the name of the parameter and the value of it, we can even see how we can combine the new Expression Language account to access other parameters that We are receiving and then propagating them to another JSP component.

These are just some of the examples of tags that we can use with JSTL. Below we will create some examples to implement the use of JSTL by combining them with our JSPs.

ONLINE COURSE

SERVLETS & JSP' S

By: Eng. Ubaldo Acosta

CURSO DE SERVLETS Y JSPS

www.globalmentoring.com.mx