JavaServer Pages (JSP)

- **Objective**: using HTML more directly with Java;
- **Comparison: Javascript versus Servlets versus JSP:**
  1) **Javascript**: client-side, it needs support and configuration in the browser. About 10% of all Internet has Javascript blocked for security reasons;
  2) **Servlets and JSP**: server-side. No action needed from the user, they run in any browser;
- **JSP**: jsp files based in HTML in which Java codes are written between <% and %> (these codes are called *scriplets).* JSP files are compiled by a JSP compiler inside Tomcat which converts a JSP file in a servlet. **Example of part of a Java code in a JSP file:**

```
<%
	String msg = "Hello World!";
	out.println(msg);
%>
```

- **Exercise 1:** Modify the project "class5" made in the last class. Modify the file "index.html" adding the following script:

```
<script type="application/javascript">
	// Example of code in JavaScript
	document.write("<br>This is a Javascript<br>");
	for(i=1; i<=4; i++){
		document.write("Line number "+i+" of my JavaScript code<br>");
	}
	alert("Finished!");
</script>
```

Example of a form validation using Javascript:
http://www.w3schools.com/js/js_form_validation.asp

- **Exercise 2:** Modify the project "class5" made in the last class. Create a JSP file named "firstjsp.jsp" with the following:

```
<html>
	<head>
		<title>First JSP</title>
	</head>
	<body>
		<%
			String msg = "Hello World!";
			out.println(msg);
		%>

		<%="Another message here!" %>

		<!-- a comment in html -->
		<%-- a comment in jsp --%>
	</body>
</html>
```

Don't forget: add a hyperlink in "admin.html" to the file "firstjsp.jsp".

- **Exercise 3:** Modify the project "class5", creating a JSP file to list all students. Name it as "listjsp.jsp". Compare it to the file JSP with the servlet that you made for the same function.

```
<body>
	<%
		StudentDAO dao = new StudentDAO();
		List<Student> students = dao.getList();

		for ( Student student : students){
			out.println("Name: "+student.getName()+" Email:
		"+student.getEmail()+" Address: "+
	student.getAddress()+"<br>");
		}
	%>
</body>
```

Don't forget to add the *imports* and to restart Tomcat server before checking the changes in the browser:

```
<%@ page import="java.util.*, br.edu.ufabc.prograd.dao.*,
br.edu.ufabc.prograd.model.*"%>
```

- **Advantage of using Javabeans**: is that, in JSP, we can instantiate classes if the constructor of the class is public and with no arguments. Besides, the Expression Language automatically understands that ${student.name} means student.getName() and, that's why, is important to obey the conventions regarding Javabeans (notice that the parameter "name" in ${student.name} is in lower case). Many other tools are also based in Javabeans such that Hibernate, Structs, VRaptor, JSF, EJB, etc. For this reason, in our project, we can instantiate inside HTML an object of the type student as follows (add this code inside the tags <body></body> in HTML).

```
<jsp:useBean id="student" class="br.edu.ufabc.prograd.model.Student" />
I am printing: ${student.name}
```

TAGLIBS

- **problem with JSP**: abuse of Java code inside JSP files;
- **solution proposed by Sun**: set of tags (*tag library* or *taglib*) to replace parts of code;
- **Taglib recommended by Sun:** JSTL (JavaServer Pages Standard Tag Library);
- Download the libraries of JSTL (available in: http://jstl.java.net/) and copy to the folder lib;
- **Exercise 4:** Modify the project "class5", creating a JSP file to list all students. Name it as "listtaglib.jsp". Import the taglib JSTL adding in the beginning of the file the following:

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

- The prefix replaces the obligation of writing the entire uri each time we need to use one tag of the library. Add the code to list the students:

```
<jsp:useBean id="dao" class="br.edu.ufabc.prograd.dao.StudentDAO" />
<c:forEach var="student" items="${dao.list}">
      Name: ${student.name}<br>
      Email: ${student.email}<br>
      Address: ${student.address}<br><br>
</c:forEach>
```

- **Exercise 5:** Modify the file "listtaglib.jsp", adding the data of the students in a table HTML with the titles of the columns in bold and with the color of the lines alternating between blue and yellow.
- **Exercise 6**: In the file "listtaglib.jsp", make a validation for the field email. If the field is not empty, add a hyperlink to open the software for emails of the computer so that the user can send a message for that student:

```
<c:if test="${not empty student.email}">
      <a href="mailto:${student.email}">${student.email}</a><br>
</c:if>
```