COMP4711 Assignment 2 (Winter 2009)
Due Dates: Mar 31, 2009 23:59

Soccer League Database

Background

You are now (finally) ready to add some relevant content to your soccer webapp. Enhance your web application as follows, using the same pairs as for assignment 1.

(1) JSP to present the ***league list***
(2) XML document and presentation for your ***team's roster***
(3) XML document and presentation for your ***team's schedule***

The **CFLmodel.zip** file on shareout contains some server-side model classes that were used last year. They might be useful this year, though there is no requirement to incorporate them.

Details follow.

League List *(league.jsp)*

You want to modify your **league.jsp** page to generate a team list dynamically, with links to all of the other projects.

The data for this list is in the **4711teams.xml** document, in the labs folder on share-out. Each team's ID is the value that you want to use as a link for the other teams' webapps. You want to modify your webapp configuration so that you use your team ID as the context name, so that everyone's links work properly.

The league list should include the name of each team and its members.
The list could be presented as an unordered list, or as a table.
The list presentation might include the team ID explicitly.
The team's league should be a hyperlink to their website.
The list could be sorted by ID or by league name.

Example:

```
   <team id="B1">

      <members>Alex F &amp; Clay T</members>

      <league>Belgium Jupiler</league>

      <url>http://www.footbel.be/en/test.html</url>

      <url>http://www.sport.be/fr/jupilerproleague/</url>

   </team>
```

This team's context is “B1”, meaning that their website would be linked to as “/B1”.
This team's entry in the league list might look like

- Belgium Jupiler (Alex F & Clay T)

In your list, have each team's league name should link to a webapp on the same host and port that you are being served from, but with a webapp context that comes from the data (i.e. that team's id).

The list itself will form part of the league.jsp page – specifically, the content or body. You may use JSP directives <%@ ... %>, declarations <%! ... %>, expressions <%= ... %> and scriptlets <% ... >, but you may not use JSP tags <JSP:... /> other than JSP:include.

Team Roster *(index.jsp)*

You want to make a DTD-constrained XML document to contain your team's roster. For each player, you will need their jersey number, name and position. You may include additional information if you like (import or not, height, weight, birthdate, college).

You have two teams (one each) ... each should be in their own XML file, and both should be constrained by the same DTD.

The DTD used should be external, and located in the content root of your webapp (the web folder?) so that it can be referenced by the user's browser. Though not the best practice, you could place the XML documents there too, but a better practice is to put them under the WEB-INF folder, perhaps in a “data” subfolder.

Modify your **index.jsp** page so that (1) it is more relevant to your team (no filler text) and (2) so that it includes the rosters you have built. You will need an XSL stylesheet to transform the roster documents into HTML, and your JSP will have to perform server-side transforms to make them into readable HTML.
Hint: Lab 9!

If you want to make the home page more readable, you **could** put the roster on a separate page, but that needs to be clear when I look at the home page. An alternate might be to use a tabbed component so that both rosters are there but only one visible at a time.

Team Schedule

You want to make a schema-constrained XML document to reflect your team's schedule and history. You want to have dates, opponents, locations and scores. This data will come from the schedule page for your team, eventually, but for now manually add data for the March games for your team. The schedule data will include historical data up to the 17th, and planned games for the 18th on.

You have two teams (one each) ... each team schedule should be in its own XML file, and both should be constrained by the same schema.

The schema used should be external, and located in the content root of your webapp (the web folder?) so that it can be referenced by the user's browser. Though not the best practice, you could place the XML documents there too, but a better practice is to put them under the WEB-INF folder, perhaps in a “data” subfolder.

Modify your **history.jsp** page so that it includes the history (played games only) you have built. You will need an XSL stylesheet to transform the schedule documents into HTML, and your JSP will have to perform server-side transforms to make them into readable HTML.
Hint: Lab 9!

Modify your **schedule.jsp** page so that it includes the schedule (unplayed, i.e. “future” games only). You will need an XSL stylesheet to transform the schedule documents into HTML, and your JSP will have to perform server-side transforms to make them into readable HTML.

If you want to make these page smore readable, you **could** put the schedules/history on separate pages, but that needs to be clear when I look at the home page. An alternate might be to use a tabbed component so that both teams are there but only one visible at a time.

Assignment Tasks

1. Implement the above (duh).
2. Make sure your webapp is configured to use Tomcat and not Glassfish, please.
3. Modify your webapp properties so that the context name used matches the team IDs mentioned above. I want to display all of the webapps at the same time, so you can see your classmates' work -> this needs unique webapp contexts.
4. Zip up your netbeans project and submit it to share-in, using a suitable name, like ParryJimComp4711Ass2.zip.

Marking Guideline

This assignment will be marked out of 20 ...

- 2 marks for the team list presentation, integrated into league.jsp
- 2 marks for the roster XML documents
- 2 marks for the roster DTD
- 2 marks for the roster XSL
- 2 marks for index.jsp integration of the rosters
- 2 marks for the schedule XML documents
- 2 marks for the schedule schema
- 2 marks for the schedule stylesheet
- 2 marks for schedule.jsp and history.jsp enhancements
- 2 marks for integration and consistency of the resulting webapp.