COMP4711 Assignment 3 (Winter 2009)
Due Dates: April 18, 2009 23:59

Soccer League Processing

Background

You are now ready to apply some of the other techniques we have talked about and practiced, in this final version of your fan website.

Enhance your web application as follows, using the same pairs as for assignment 1.

(1) JSP to **prompt** for the URL of a team's online scores.
(2) Servlet to extract those online scores and **convert them** into an XHTML document.
(3) Servlet to process the XHTML document, and recreate or update your existing **history and schedule** data.
(4) Interactive page to **predict** the outcome of your two teams playing each other.

There are several files in share-out that you will find useful for this:

- **winnipeg.zip,** in examples, is an assignment 3 project from last year. It uses the tagsoup package to handle mal-formed HTML pages
- **tagsoup-1.2.jar**, in tools, is a Java package to handle ugly HTML. It does not actually convert anything – it provides a SAX parser that works even on mal-formed HTML. The jar should be put inside the lib folder of your webapp, for wasy reference. This package is similar in function to HTMLTidy
- **tagsoup-1.2-src.zip**, in tools, is the source code for the tagsoup library above. There are a bunch of caveats about building it with JDK5 or JDK6, and a dependency on a third party tool, but the source code might prove instructive.

Details follow.

Score Prompt *(update.jsp)*

The **update.jsp** page needs to present a form with two text fields, one each to hold the URL of your teams' schedule HTML page. Provide a default value for the text field, matching the input page/file URL which was correct at the time of assignment submission.

If you prefer, you can have two forms (on the same HTML page), each of which addresses a single team, or you can handle each in a separate “area” or column for the respective teams.

You will need to determine the default values for these pages manually, and then store them as webapp parameters in your project (hard-coded names would be a no-no).

The URLs may change over time, but hopefully they will be valid for the next few weeks :)

The action of your form should map to the servlet described in the next section.

HTML conversion *(servlet)*

The update.jsp form should have a POST to a servlet in your webapp which will **process the supplied file** by converting it from potentially malformed HTML into XHTML, producing a new .xml file stored by your webapp. This servlet will then forward the request to a second servlet, which will update your original schedule.

There are not a lot of cross-platform packages to handle malformed HTML. The most popular is HTMLTidy, but it is a Windows program. The only cross-platform package we have found is “tagsoup”, a copy of which is in the COMP4711 tools folder.

The “winnipeg” project from last year uses tagsoup. If you take a look at its code, you will see that tagsoup provides a SAX parser, which can be used to build a well-formed XML document, which you then save conventionally.

This has been set as a separate servlet and assignment component, because it is qualitatively distinct from the other tasks.

On conclusion of the HTML document conversion, this servlet can forward the request to the next in line by using a RequestDispatcher.

Schedule Update (servlet)

This servlet needs to process the XML file produced by the previous step, and either update your existing schedule XML document, or produce a new one. When done, it should redirect the request to your webapp's home page.

Updating the original schedule document would involve adding scores for any games that have not been played, according to the original data. The alternative is to re-generate the schedule XML document from scratch, using the document produced in the previous step.

When the dust settles, the schedule documents should reflect all of the played and unplayed games for the team(s) that they apply to.

The redirection can be handled by a RequestDispatcher.

Predictor (predict.jsp)

This page needs to prompt the user for the team competed against, and predict the outcome.

You know your team(s), and have its (or their) history.

For each of your teams, you want to provide a form which lets the user pick from the other teams that yours has played (defaulting to the other one in your project), and use AJAX to send a request to a new servlet in your webapp, with the two teams as parameters, then update a div with the id of “result”, in this page, with the response from your servlet.

Major hint: the two team predictions may not be the same (in terms of winner and predicted score) ... see below.

The servlet that the AJAX code wil connect to show take two parameters (team1 and team2, for instance), and predict the result.

Team1 will be deemed to be the away team, and team2 the home team.

Predict the score for each team as 75% of the average goals that team has scored against their opponent in previous history, and 25% of that team's average score in league play todate.

The prediction might look something like “Whitecaps 5 vs Sounders 1” or “Whitecaps at Seattle, 5:1”.

Assignment Tasks

1. Implement the above (duh).
2. Zip up your netbeans project and submit it to share-in, using a suitable name, like ParryJimComp4711Ass3.zip.

Marking Guideline

This assignment will be marked out of 20 ...

- 2 marks for the score prompt, prompt.jsp
- 4 marks for the HTML -> XML conversion servlet
- 2 marks for the XML history update/re-generation servlet
- 4 marks for the prediction page with AJAX, predict.jsp
- 4 marks for the prediction calculator servlet
- 2 marks for validation of the converted XML and updated history
- 2 marks for integration and consistency of the resulting webapp.