Introduction

The purpose of this homework is to get you acquainted with the server side of the project plan.

We have provided you with a zip file for enabling and running your own instance of Tomcat, the servlet container. We have also provided a sample web application containing one example servlet in two formats. The binary distribution contains the binary application file, associated deploy scripts and release notes only. The source distribution contains all the files needed to build and deploy the application.

You have two tasks for this homework.

1. Get your individual instance of Tomcat up and running on one of the servers adelie or rockhopper.

2. Implement a servlet that produces a simple list of image urls available from the application.

Preparation

1. If you are working on a non-lab machine, install the course software as described on the class web site software page. Note that you need access to the tomcat jar files on the development system you are using. This does not have to be the same as the system where the Tomcat server is actually running.

2. Copy the tomcat4-instance.zip file from the web site to your account on rockhopper or adelie (either one is okay). Unzip that file in your home directory there, and then follow the instructions in the release notes for getting Tomcat running. Once you have done that, then continue to step 3.

3. Download the two distribution zip files. Unzip the source archive on your development system. You need to update the file build.properties so that catalina.home points to the location of the tomcat installation on your system. Read the servlet development section of the class software page for more information about this.

4. At this point you should be able to type “ant” at the top level spsb directory and ant will compile the code (Contexter.java) and build a new web archive file spsb.war in build/bin.

5. Unzip the binary archive on the server system (rockhopper or adelie). Note that there is a directory spsb/build/bin. This is where the spsb.war file resides.

6. Edit the deploy.properties file so that it knows how to talk to your copy of Tomcat. All that is required is that you update the manager.url line so that instead of 8100, it uses 81xx, where xx is the instance number you have been assigned and are using for your copy of Tomcat.

7. At this point you should be able to type “ant –f deploy.xml install” at the top level spsb directory and install your web application. You should be able to go to URL

   http://rockhopper.cs.washington.edu:81*xx*/*username*/context

   and see a dump of various little bits of servlet related info. Also you should be able to go to URLs like

   http://rockhopper.cs.washington.edu:81*xx*/*username*/images/ant.png

   and see the images that are in the war file.

Servlet Development

Your task for this assignment is to write a servlet called Lister that creates a list of image names and URLs from which to read them. This servlet executes in response to an html request like this:

http://rockhopper.cs.washington.edu:81*xx/username*/list

and produces a list like this:

Ant$http://rockhopper.cs.washington.edu:81*xx*/*username*/images/ant.png
Bird$http://rockhopper.cs.washington.edu:81*xx*/*username*/images/bird.png
Cat$http://rockhopper.cs.washington.edu:81*xx*/*username*/images/cat.png

1. Read the file spsb\web\WEB-INF\web.xml. It shows how to define a servlet including its name, class, init-params, and mapping to a url pattern.

2. Write a class Lister that is mapped to a url similar to the one shown above and produces an appropriate list of names and urls. By appropriate, I mean that the host, port number, and context path in the list of urls should always be correct, no matter where the application is installed.

3. Build and install your web application with your instance of Tomcat.

4. Update the release notes in spsb\doc for your application.

Turn in

Using the Ant target “dist” build binary and source archive files for your web application. Verify that the archives have the expected files, then follow the turnin link on the web site and turn in the two files.

This homework is due before midnight Tuesday April 15th. Don’t be late, the turnin server closes automatically.