Assignment #8b

DataEntry servlet

Filenames: DataEntry.html, DataEntry.java

Create an HTML form that requests data from the user. Submit that form data to a servlet. For output, display the user's submitted data back to the user. For extra experience, also save the data in a server-based text file.

1. Create your HTML data entry form, and store it in your tomcat/webapps/ROOT directory. You can access it at:

http://mislab.business.msstate.edu:8080/DataEntry.html

(Of course, substitute your primary port number.)

2. Your form tag should refer to your servlet in its action attribute:

```
/servlet/abc123.DataEntry
```

3. In Eclipse, create a new Java project. Name it Servlets. You will not actually run your servlets in Eclipse, so we suggest that you simply use this one project for the rest of your Java programming assignments. Just create new classes as you need them, and keep them all in this project. You will upload them as needed from the project into your mislab account.

4. This project needs some non-standard classes that are not automatically included in an Eclipse project. Download the file servlet.jar from our course web site. This is a java archive file that contains several servlet-related classes.

5. In Eclipse, right-click (or control-click) on your project name, then select Properties. This will open the Properties Window.

6. In the Properties Window, select Java Build Path along the left, then click the Libraries tab.

7. Click the "Add External JARs" button, and add servlet.jar to your project.

8. Create a class named DataEntry.

This class should be a subclass of HTTPServlet, not of java.lang.Object.

Click the Browse button. In the pop-up window, start typing the word httpser. When you see the HTTPServlet class listed, select it as your superclass.

9. Eclipse will generate an import statement:

```
import javax.servlet.http.HttpServlet;
```

We recommend changing this to:

```
import javax.servlet.http.*;
```

10. Add another import, for I/O exception handling:

```
import java.io.*;
```

11. Write a method that will be called automatically to respond to an HTTP get request:

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
throws IOException
{
}
```

12. Your servlet needs to send back a "doctype" indicator before it sends any other output. You can do this with your response parameter, which has a setContentType() method.

response.setContentType("text/html");

13. You can use your response parameter's getWriter() method to get a PrintWriter object, which you can use to send text back to the browser.

```
PrintWriter out=response.getWriter();
out.println("This is a test.");
```

14. Save your work in Eclipse, then upload your DataEntry.java file to tomcat/webapps/ROOT/WEB-INF/classes.

15. Telnet to your account and move to your classes directory. You need to compile your Java code on mislab, because tomcat does not yet support Java 1.8 byte code. mislab has Java compilers for Java 1.4, 1.5, 1.6, 1.7, and 1.8. In your classes directory, type:

```
javac DataEntry.java
```

16. Move to your tomcat/webapps/ROOT/WEB-INF directory.

For security purposes, your tomcat server will not serve any servlet that is not listed in your web.xml file.

Use pico to edit web.xml. You will see entries in the file for HelloWorldExample and SnooperServlet. You need to put an entry in for your DataEntry servlet.

```
<servlet>
      <servlet-name>DataEntry</servlet-name>
      <servlet-class>DataEntry</servlet-class>
</servlet>
<servlet-mapping>
      <servlet-name>DataEntry</servlet-name>
      <url-pattern>*.DataEntry</url-pattern>
</servlet-mapping>
```

Copy/paste in pico is a bit tricky, but here's how you do it. Move to the first <servlet> container for SnooperServlet. Press Shift-Control-6. At the bottom of the screen, you will see "[Mark Set]".

Arrow-down to highlight the lines you want to copy.

Press Control-k.

This will cut the highlight block, so immediately press Control-u to paste it right back where it was. Now you can Control-u again to paste it again, and have a copy that you can edit. Change every SnooperServlet to DataEntry.

VERY IMPORTANT: Any mistake in this file will prevent your tomcat server from working. If you make any mistake in this file, you will probably get a blank screen response from your tomcat server when you try to access any page.

17. You will need to stop your tomcat server and start it again, so it will process your web.xml file.

`stoptom`

`starttom`

18. Modify your servlet to retrieve the submitted form data from your HTML form. If you have a form field named lastName, for instance, you can retrieve the form data with:

```
String lastName=request.getParameter("lastName");
```

Echo information back to the browser to verify that you can indeed retrieve the submitted form data.

Remember that any time you edit your .java servlet, you must:

- save your work in Eclipse (to generate a new .java file)

- upload the .java file from your workspace to your classes folder

- switch over to your mislab account (telnet) and compile the .java file

- stop your tomcat server

- start your tomcat server

19. If you can get one more step done, you will be in really good shape. Write the submitted data into a server-based file (the file will be in your classes folder).

```
try
{
  FileWriter file=new FileWriter("DataEntry.txt",true);
  file.write(lastName+"\t"+firstName+"\n");
  file.close();
}
catch(Exception e)
{
}
```

20. In your index.html, change the href in the anchor tag for this assignment to request DataEntry.html. Unlike previous assignments, you will not display your Java source code now, because you now have a working web-based assignment. We want the DataEntry assignment to work, not simply display source code.

Created 2019-01-09 23:51:01 UTC by Aaron Kimbrell
Updated 2019-01-10 16:40:22 UTC by Aaron Kimbrell