# SEGMENT 2

"Fear of failure must never be a reason not to try something." - Frederick Smith

# **Overview:**

In **Segment 2 (your second coding project)** you will now *convert most of the html files you created in* Segment 1 to JSP pages. This will enable the pages to interact with the database and extract information to be displayed to the user.

# **Before Starting:**

Before starting Segment 2 make sure you have the following installed and ready:

- 1. Latest Java SE Development Kit (J2SE SDK)
- 2. Latest Tomcat webserver
- 3. Eclipse Helios
- 4. ODBC Connection

# **Preparation:**

If you have not used Eclipse Helios to develop your HTML pages please do the steps below. If you have already used Helios it is also good to review the steps below to confirm that you're Eclipse Helios is ready for use in Segment 2.

#### 1. Open Eclipse:

Go to your Eclipse folder and double click on the Eclipse icon.

# 2. Select your Workspace:

The "Workspace" dialogue box will open.

Select a wi Eclipse store Choose a wo	orkspace : your projects in a folder called a workspa rkspace folder to use for this session.	ce.		
Workspace	Gi, Visers Valence in Billion of Caroling Store		•	Browse
🖾 Use this at	the default and do not ask again			
⊡ <u>V</u> se tris ai	the detault and boindt ask again	ОК		Cancel

**Remember:** Your workspace is where all your project files are kept.

#### 3. Create a new Dynamic Web Project:

The Eclipse workbench should now be open. Click on File  $\rightarrow$  New  $\rightarrow$  Project

File	Edit Navigate Search Project Run Window	Help	
	New Alt+ Open File	-Shift+N ▶	IPA Project           Image: Constraint of the project           Image: Constraint of the project
	Close Close All Ctrl+	Ctrl+W Shift+W	Dynamic Web Project EJB Project EJB Project
	Save Save As Save All Ctri-	Ctrl+S +Shift+S	Application Client Project     Static Web Project     Project
8	Move Rename Refresh Convert Line Delimiters To	F2 F5	Serviet     Session Bean (EJB 3.x)     Message-Driven Bean (EJB 3.x)     Entity     Web Service

4.	<ol> <li>Select the project type:</li> </ol>		
	Select "Dynamic Web Project" then click on		
	next.		

**REMEMBER:** Since we are using JSP we need **a** dynamic web project.

	New Project	
on	Select a wizard Create a Dynamic Web project	
	Wizards:	
	type filter text	
eed <b>a</b>	<ul> <li>b b Eclipse Modeling Framework</li> <li>b b ElB</li> <li>b b Java</li> <li>b b Java EE</li> <li>b JavaScript</li> <li>b b JPA</li> <li>b b Plug-in Development</li> <li>b b Veb</li> <li>b Dynamic Web Project</li> </ul>	E
	G Stadt Web Project G Web Fragment Project ▷ ➢ Examples	
	? < Back Next > Finish	Cancel

#### \_\_\_\_ 🗆 🗾 📈 New Dynamic Web Project 5. Fill in the project options for: Dynamic Web Project - Project Name: myWeb 67 Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application. - Target Runtime: Apache Tomcat 6.0 Project name: - Dynamic web module version: 2.5 Project location Use default location Location: C:\Users\taleweaver\workspace Browse... Target runtime Apache Tomcat v6.0 New Runtime... Click on "Finish" Dynamic web module version 2.5 -Configuration Default Configuration for Apache Tomcat v6.0 ▼ Modify... You now have a new Dynamic Web project! A good starting point for working with Apache Tomcat v6.0 runtime. Additional facets can later be installed to add new functionality to the project. EAR membership Add project to an EAR EAR project name: EAR ▼ New Project... Working sets Add project to working sets \* Select... Working sets: ? < Back Next > Finish Cancel

**NOTE:** Now you have a new dynamic web project. Always remember when you add a new HTML or JSP file to put them in the "webcontent" folder. *They will not work if placed in other folders.* 



# 6. Add pre-coded class files.

In a real-world programming project you are not required to code the entire project most of the time. The pre-coded class files contain database access and business logic needed by the application so that you will no longer need to do those parts. Your major role in this project is to code the display logic via JSP technology.



X web.xml

b. Right-click on WEB-INF and select IMPORT...



c. Select "File System" under the "General" Group



Import resources from the local file system into an existing project.	r <sup>2</sup> s
Select an import source:	
type filter text	
a 🗁 General	
Archive File     Suidan Designs into Washington	
<ul> <li>Existing Projects into Workspace</li> <li>File System</li> </ul>	
Preferences	
CVS CVS	
D Con Java FF	-
Plug-in Development	
Remote Systems	
Environmentaria	
p 👉 lasks	
> 🕞 Web	
N Pa Web services	-

d. Browse for "xml" folder found in "School Files"

Import from directory		
Select a directory to import from.		
Folder: Xml		

e. Select the "web.xml" file. Take note: select the exact file, not the folder.

j Import	
File system Import resources from the local file system.	
From directory: C:\stuff\ADIS-SMART\School Files\mb	·
	Select this
×	
Filter Types Select All Deselect All	
Into folder: myWeb2/WebContent/WEB-INF	Browse
Options Overwise existing resources without warning Overwise complete folder structure Create selected folders only Advancede>	
c Back Nett > Finish	Cancel

f. Eclipse will ask you if you want to OVERWRITE the existing XML File. Click on "Yes".



g. Go to "Project Explorer." Double-click on "web.xml." This will open the file in the coding window. You should now have something similar to the one in the picture.



7. Now that we have added the proper xml file we add the pre-coded classes. In "Project Explorer" right click on "WEB-INF", select "NEW" and then "FOLDER". Name the new folder "classes"



- ) Java EE myWeb2/WebContent/WEB-INF/web.xml yEclipse File Edit Navigate Search Project Run Design Window Help 📩 • 🗐 🖗 🚔 🔅 • 🝳 • 💁 • 3-6- 00 4- 0 4 Project Explorer x web.xml <?xml version="1.0" encoding 8 8 8 myWeb
   myWeb2
   JAX-WS Web Services
   Deployment
   Mava Resource <!DOCTYPE web-app PUBLIC "-/ ⊖<web-app> ⊖ <servlet> New Bay Resource
   Bay Jave Schorte
   Bay Jave Schorte
   Bay Schorte
   Bay MetContext
   Bay MetContext
   Bay MetContext
   Bay MetContext
   Copy Qualified Name
   Bayte
   Copy Qualified Name
   Bayte
   Detet
   Bayte
   Detet
   Bayte Alt+Shift+W + Ctrl+C Ctrl+V Delete Servers
   b b test Remo Ctrl+Alt+Shift+Down Build Path Move... tn Rename... F2 Ma Import... 🛃 Export... 8 Refresh F5 Validate h.
- 9. Select "File System" under the "General" Group.



10. Browse for "classes for Access" folder found in "Access Version" under "School Files"

le system ource must not be empty.	
rom directory:	▼ Browse
	Import from directory
Filter Types	School Files     Cases version     Connections     Connections     Connections     Connections     Connections     Connections     Connections
3	Next > Einich Canad

8. Right click on the "classes" folder and select "import."

11. Select all the ".jsp" files. Take note: select the exact file, not the folder. Do this for all the folders found under "classes for Access"

Import			
File system Import resources from the local fil	le system.		
From directory: C:\stuff\ADIS-SM	MART\School Files\Access versio	on\classes for Access 👻	B <u>r</u> owse
<ul> <li>classes for Access</li> <li>beans</li> <li>beans&lt;</li></ul>	V L Su	yscriberBean.class DscriberBean.java	
Filter <u>Types</u> <u>Select All</u> Into folder: myWeb2/WebConte	Deselect All		Browse
Options Querwrite existing resources w Greate complete folder structu Create selected folders only Advanced >>	ithout warning Ire		
?	< <u>B</u> ack <u>N</u> ext	> <u>F</u> inish	Cancel

Click "Finish" once you have selected all files.

**12.** Go to Project Explorer you should now see the directory structure pictured below.



You are now ready to begin coding Segment 2.

# CODING SEGMENT 2 (SOME HINTS TO START YOU OFF)

If you did SEGMENT 1 well then it's just a matter of copying your HTML content into their corresponding JSP files. After you have done this you need to add the JSP logic that will make the page run dynamic content.

**NOTE:** Read the *technical specifications* section of the Segment Binder to answer the question "What do I need to code?"

# Input\_form.html

The "input form" does not need to be converted to jsp since the form itself will do no processing. Make sure the target for the submit button correctly follows the specifications. The code will look something like the code below:

Sample code for input\_form.html (without layout/page decorations)

```
<html>
<html>
<head>
<title>Account Number Input Form</title>
</head>
<body>
<form method="post" action="controller">
Subscriber Account Number:
<input name="account_number:
<input name="submit" type="text" />
<input name="page" value="subinfo"type="hidden" />
<input name="submit_button" type="submit" value="Submit" />
</form>
</body>
</html>
```

**NOTE: Re-check your input\_form.html so that it conforms to the form elements here. Take note of the** hidden input control.

# Subscriber\_info.jsp

Make sure you read the pseudocode (Green text in technical specifications) for this very carefully. Sample code snippets below with reference to the pseudocode:



# Continuing the code:

```
<html>
<head><title>Subscriber Info</title></head>
<body>
  Subscriber Name:
       <tjsp:getProperty name="subscriberInfo" property="subscribername"/>
     MSISDN:
       jsp:getProperty name="subscriberInfo" property="MSISDN"/>
     IMSI:
       <jsp:getProperty name="subscriberInfo" property="IMSI"/>
     Account Number:
       jsp:getProperty name="subscriberInfo" property="account_number"/>
```

#### Pseudocode:

🔁 ADIS	- Segment 2 - Specific	cations Binder.pdf - Adobe Reader	x
File E	dit View Document	t Tools Window Help	×
8	🄬 -   🌍   👍	😼 32 / 35 🕑 🖲 139% - 🙀 🚼 Find -	
		<pre><html>     <head>     <title>Subscriber Information</title>         link to the smart stylesheet         </head>         <body>         Include the header.html file          In the first column display the following labels: subscriber         name, MSISDN, IMSI, account number, birthday and mother's maiden         name.</body></html></pre>	^
		In the second column invoke the methods of the bean to retrieve the corresponding subscriber information for this instance and populate the table. <hr/>	

# Final Part of the Code:

```
<hr>
   <form method="post" action="controller">
        <input name="page" value="input" type="hidden"/>
        <input value="Input Form"
               type="submit"/>
   </form>
   <form method="post" action="controller">
        <input name="account number"
               value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
               type="hidden"/>
        <input name="imsi"
               value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
               type="hidden"/>
        <input name="page" value="subusage" type="hidden"/>
        <input value="Get Subscriber Usage"
               type="submit"/>
   </form>
   <form method="post" action="controller">
        <input name="account number"
               value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
               type="hidden"/>
        <input name="imsi"
               value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
               type="hidden"/>
        <input name="page" value="mosubusage" type="hidden"/>
        <input value="Get Subscriber Monthly Usage"
               type="submit"/>
   </form>
   <form method="post" action="/web/controller">
        <input name="account number"
               value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
               type="hidden"/>
        <input name="imsi"
               value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
               type="hidden"/>
        <input name="page" value="ir" type="hidden"/>
        <input value="Enable International Roaming"
               type="submit"/>
   </form>
   </body>
</html>
```

#### Final Pseudocode for subscriber\_info.jsp:



**NOTE:** You should now be able to finish Segment 2.

**NOTE:** The code above only completes the logic portion of the pseudocode. The layout parts are not shown.

**NOTE:** Complete Code (logic portion) for subscriber\_info.jsp:

```
<jsp:useBean id="subscriberInfo" scope="request" class="beans.SubscriberBean">
   <jsp:setProperty name="subscriberInfo" property="account number" />
</jsp:useBean>
<%@ page errorPage="/sub info no data.html" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
 <head><title>Subscriber Info</title></head>
 <body>
   Subscriber Name:
           <jsp:getProperty name="subscriberInfo" property="subscribername"/>
       MSISDN:
           <jsp:getProperty name="subscriberInfo" property="MSISDN"/>
       <t.r>
           IMSI:
           <jsp:qetProperty name="subscriberInfo" property="IMSI"/>
       Account Number:
           <jsp:getProperty name="subscriberInfo" property="account number"/>
       </t.r>
   <hr>
   <form method="post" action="controller">
       <input name="page" value="input" type="hidden"/>
       <input value="Input Form"
              type="submit"/>
   </form>
   <form method="post" action="controller">
       <input name="account_number"
              value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
              type="hidden"/>
       <input name="imsi"
              value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
              t.vpe="hidden"/>
       <input name="page" value="subusage" type="hidden"/>
       <input value="Get Subscriber Usage"
              type="submit"/>
   </form>
   <form method="post" action="controller">
       <input name="account number"
              value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
              type="hidden"/>
       <input name="imsi"
              value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
              type="hidden"/>
       <input name="page" value="mosubusage" type="hidden"/>
       <input value="Get Subscriber Monthly Usage"
              type="submit"/>
   </form>
   <form method="post" action="controller">
       <input name="account number"
              value="<jsp:getProperty name="subscriberInfo" property="account number"/>"
              type="hidden"/>
       <input name="imsi"
              value="<jsp:getProperty name="subscriberInfo" property="IMSI"/>"
              type="hidden"/>
       <input name="page" value="ir" type="hidden"/>
       <input value="Enable International Roaming"
              type="submit"/> </form>
   </body>
</html>
```

# **RUNNING THE SITE**

Once you have coded *input\_form.html* and *subscriber\_info.jsp* you can test your work by clicking on the project folder and then clicking on the go button (green arrow) on the toolbar.

The *web.xml* file tells Tomcat to display *input\_form.html*.

Enter the value 119777259. This will display the subscriber info for Verlin Samonte. If you get his information then you have coded the two files correctly.

Other test data are found at the end of the Segment 2 Test Script.

# **TESTING SEGMENT 2**

Segment 2 is the first segment that comes with a formal testing document. To complete the segment and to check if you have coded things correctly follow the Segment 2 test script and fill it out completely.

You should fill out the columns for actual result, test result and action taken. Actual results is what you observe (usually a restatement of expected result if it is the same), test result is passed if it actual results match expected results. Action taken is filled out if Test Result is failed. Usually Action taken contains the entry "Recoded module" or "Fixed coding error."

"What is genius, anyway, if it isn't the ability to give an adequate response to a great challenge?" -- Bette Greene

#### **REMINDERS:**

If you think some of the steps are missing please refer to segment 1 and segment 2 binders.

#### Things to submit for Segment 2:

1 html file: corrected input\_form.html

3 JSP files:

- subscriber\_info.jsp
- subscriber\_usage.jsp
- subscriber\_monthly\_usage.jsp

1 filled-out Test Script

# **Deadlines:**

End of class (March 19):

- input\_form.html and subscriber\_info.jsp
- email to: <u>rlramos@gmail.com</u> subject: Appdev Segment 2 A 3<sup>rd</sup> term 2010-11 <your name>

End of day (March 27, 11:59pm):

- subscriber\_usage.jsp and subscriber\_monthly\_usage.jsp
- filled out test script
- email to: <u>rlramos@gmail.com</u> subject: Appdev Segment 2 B 3<sup>rd</sup> term 2010-11 <your name>

rlramos 2011