Introduction to Web Applications using JSP, Servlets, Jstl and JQuery
Workshop for ICOM5016

Harold Valdivia Garcia

University of Puerto Rico, Mayaguez Campus

October 24, 2010
Outline

1. MediaDB: A ICOM5016 project
2. Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery
3. Setting Eclipse and Tomcat
4. Examples
   - WebApp Calculator
   - WebApp University
5. Deploying a Web App to Tomcat
Outline

1. MediaDB: A ICOM5016 project
2. Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery
3. Setting Eclipse and Tomcat
4. Examples
   - WebApp Calculator
   - WebApp University
5. Deploying a Web App to Tomcat
MediaDB: A ICOM5016 project

Requirements
Setting Eclipse and Tomcat
Examples
Deploying a Web App to Tomcat

MediaDB: A ICOM5016 project

Browse it here mediadb

Music | Video

Songs | Albums

Genre
Salsa
Pop
Latin Pop
Reggae
Reggaeton
Merengue
Romantic
Heavy Metal
Classic
Compilations

Browse Movies
Drama
Adventure
Comedy
Horror
Western

Music

Search: Artist

A Dynasty from the Ice
by Giorgio Estefan, Amaral,

Essential Jerry Reed
by Cameron Diaz, Jesus Vazquez,

Learn To Live
by Antonio Banderas, Tom Cruise,

Sleepless Nights

Top 10
Learn To Live
Essential Jerry Reed
Doomsdayers Holiday
Sleepless Nights
Chris Knight
Appeal To Reason
A Dynasty from the Ice
Bare Sinner
Only By The Night
Life Is Not A Wasting

New Releases
The End
A Dynasty from the Ice
Sleepless Nights
Learn To Live
Essential Jerry Reed

Harold Valdivia Garcia
This presentation was made using LaTeX and Beamer
Outline

1. MediaDB: A ICOM5016 project

2. Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery

3. Setting Eclipse and Tomcat

4. Examples
   - WebApp Calculator
   - WebApp University

5. Deploying a Web App to Tomcat
Download Tomcat 6.0 from Apache-Tomcat-Project
Unzip the file into your directory "Tomcat-Dir"
Jstl (JSP Standard Tag Library) and JQuery

- Jstl is a collection of tags that implements programming structures. Download Jstl from [here](#).
- JQuery is a Ajax Library. Download from [here](#).
Outline

1 MediaDB: A ICOM5016 project

2 Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery

3 Setting Eclipse and Tomcat

4 Examples
   - WebApp Calculator
   - WebApp University

5 Deploying a Web App to Tomcat
Setting Eclipse and Tomcat

- Eclipse → Preference → Servers → Runtime Environments → Add
- Set the tomcat directory
Outline

1. MediaDB: A ICOM5016 project
2. Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery
3. Setting Eclipse and Tomcat
4. Examples
   - WebApp Calculator
   - WebApp University
5. Deploying a Web App to Tomcat
WebApp Calculator

- This web application implements a very simple arithmetic calculator. The user can send two numbers and an operation (+, -, /, *) and receive the results.
- Download it from here
- Look it at here: web-calc
The Web app has the following files:

- Calculator.java
- index.jsp
- CalculatorServlet.java
- calculator-result.jsp
WebApp Calculator

index.jsp

```html
<h1>My Calculator</h1>
<form action="./CalculatorServlet" method="post">
  <table>
    <tr>
      <td><input name="operator_1" type="text"/></td>
      <td>
        <select name="operation">
          <option value="+">+</option>
          <option value="->">-</option>
          <option value="*">*</option>
          <option value="/">/</option>
        </select>
      </td>
    </tr>
    <td><input name="operator_2" type="text"/></td>
    <td><input value="calculate" type="submit"/></td>
  </table>
</form>
```
WebApp Calculator

CalculatorServlet.java

```java
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // Get the input from the request
    Integer operator_1 = Integer.parseInt(request.getParameter("operator_1"));
    Integer operator_2 = Integer.parseInt(request.getParameter("operator_2"));
    String operation = request.getParameter("operation");

    // Calculate the expression
    Calculator calc = new Calculator();
    float result = calc.calculate(operator_1, operator_2, operation);

    // Show the results
    request.setAttribute("operator_1", operator_1);
    request.setAttribute("operator_2", operator_2);
    request.setAttribute("operation", operation);
    request.setAttribute("result", result);

    // Forward the results to the viewer
    RequestDispatcher dispatcher = request.getRequestDispatcher("calculator-result.jsp");
    dispatcher.forward(request, response);
}
```
WebApp Calculator

calculator-result.jsp

```jsp
// Get Attributes from the Request Context
Integer operator_1 = (Integer)request.getAttribute("operator_1");
Integer operator_2 = (Integer)request.getAttribute("operator_2");
String operation = (String)request.getAttribute("operation");
float result = (Float)request.getAttribute("result");

<table>
  <tr>
    <td><%=operator_1 %></td>
    <td><%=operation %></td>
    <td><%=operator_2 %></td>
    <td><%=result %></td>
  </tr>
</table>
```
Considering the MVC pattern, this web app has three components:

- Models: Courses.java, Professor.java, etc (POJOs)
- Views: index.jsp, courseViewer.jsp, etc
- Controller: Servlets + CourseMgr.java, ProfessorMgr, etc

- It uses embedded Java code
- It uses JSTL and JQuery
- Download it from here
- Look it at here: web-university
Installing JSTL

Copy the jstl jar to:
Web-name → WebContent → WEB-INF → lib
Installing JSTL

Add the jars to the classpath:
Web-project → Properties → Java Build Path → Libraries → Add Jars
Embedded Java Code in JSP Pages

- We have access to these objects: request, response, session
- We can use Java and HTML.
- Scriptlets are a Good Example of a Bad Solution.
Embedded Java Code in JSP Pages

courseViewer.jsp

```jsp
<!-- Show the courses-->

<% 
if(option.compareToIgnoreCase("SHOW_ALL") == 0){
  %>
<h2>Courses</h2>
<table cellspacing="20">
  <%
  for(int i = 0; i < coursesList.size() ; i++){
    Course course = coursesList.get(i);
  %>
    <tr>
      <td>%= course.getCid() %</td>
      <td>%= course.getCname() %</td>
    </tr>
  %>
  %>
</table>
%>
%>
```
JSTL: JSP Standard Tag Library

- We will only use the core library (http://java.sun.com/jsp/jstl/core).
- Here a good tutorial
- Do not forget to add the jstl directive in the JSP pages:

```jsp
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```
JSTL: JSP Standard Tag Library

`professorViewer.jsp`

```xml
<!-- JSTL -->
<c:if test="${option == 'SHOW_ALL'}">
  <h2>Professors</h2>
  <table>
    <c:forEach var="professor" items="${professorsList}"
      <tr>
        <td>${professor.pid}</td>
        <td>${professor.fullName}</td>
        <td>${professor.department}
          <c:if test="${professor.department == 'INEL/ICOM' }">
            <b>The Best Department!!!</b>
          </c:if>
        </td>
      </tr>
    </c:forEach>
  </table>
</c:if>
```
Here `studentViewer.jsp` using JQuery

JQuery allows us to send background requests to the Web Server.

JQuery allows us to receive asynchronous response from the Web Server.

It also manipulates the DOM.
JQuery

- You must include the jquery.js file in your pages.

```html
<head>
  <title></title>
  <meta http-equiv="Content-Language" content="English" />
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

  <link rel="stylesheet" type="text/css" href="js/style.css" media="screen" />
  <script type="text/javascript" src="js/jquery.js"></script>
  <script type="text/javascript" src="js/jsStudent.js"></script>
</head>
```
In the example, we want to add a link that retrieves the student’s details and shows them in a div tag.

```html
<!-- The js/jsStudent.js file has the jQuery handler for show_student_detail_click -->

```html
<a class="show_student_detail_click" href="./StudentWS?sid=${student.sid}" >
  get details
</a>

<h2>Student Details</h2>
```html
<div id="table-student-detail">
  <!-- jQuery -->
  It will be filled clicking the "get details" link.
</div>
```
JQuery

js/jsStudent.js

```javascript
$(document).ready(function()
{
    // register_show_student_detail_click:
    function register_show_student_detail_click()
    {
        // Get all tag with class "show_student_detail_click"
        // and set the "click-event"
        $('.show_student_detail_click').click( function(e){
                e.preventDefault();

                //Send a get-request to href
                $.get( $(this).attr("href"), function(data){
                    //Fill the html-element "table-student-detail"
                    //with the data retrieved from the service
                    $('#table-student-detail').hide();
                    $('#table-student-detail').html(data);
                    $('#table-student-detail').show("slow");
                });

            });
    }

    //Event Registration Section
    register_show_student_detail_click();
}

```

Harold Valdivia Garcia

This presentation was made using LaTeX and Beamer
Outline

1. MediaDB: A ICOM5016 project
2. Requirements
   - Your Attention
   - Web Server + Servlet Container (Tomcat)
   - Jstl (JSP Standard Tag Library) and JQuery
3. Setting Eclipse and Tomcat
4. Examples
   - WebApp Calculator
   - WebApp University
5. Deploying a Web App to Tomcat
Deploying a Web App to Tomcat

- Web-project → Export → WAR file
- Uncheck the optimize option
- Check Export source file
Deploying a Web App to Tomcat

- Tomcat installation under Ubuntu:
  - `sudo apt-get install tomcat6`

- Deploying a war file to Tomcat:
  - Copy the war file into `/full-path-tomcat/webapps/`
  - `sudo cp my-web-app.war /var/lib/tomcat6/webapps/`

- Browse your web app here:
  - `http://your-ip:8080/my-web-app/index.jsp`