Segundo Parcial - Práctica: Servlets de Java y JSP

Instrucciones

1. Este examen consiste en tareas prácticas relacionadas con servlets de Java y JSP. Para esta examen utilicen una versión mayor a la 15 por fácilidad en la utilización de los códigos, en principio también funciona para java-11.
2. Debes completar cada tarea escribiendo los archivos Java servlet correspondientes, archivos JSP, compilarlos y agregar las asignaciones de servlets al archivo `web.xml`.
3. Asegúrate de que tus servlets y páginas JSP estén implementados correctamente y produzcan la salida esperada.

Puntos

0. Aplicación Web con Tomcat

Crea la estructura de archivos para una aplicación llamada `segundo_parcial_2`, recuerda no agregar el archivo context e ir reiniciando la aplicación web app desde el manager para encontrar los errores que puedan dañar al tomcat de forma más fácil. Agrega el archivo de configuración de la aplicaición:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns = "https://jakarta.ee/xml/ns/jakartaee"
         xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation = "https://jakarta.ee/xml/ns/jakartaee
         https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd"
         version = "6.0"
         metadata-complete = "false">

    <description>Servlet Segundo Parcial - Programación 3 - ETITC.
</description>
    <display-name>Servlet Segundo Parcial</display-name>
    <request-character-encoding>UTF-8</request-character-encoding>

    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>


</web-app>
```

Y además agrega un archivo `index.html` sencillo para la aplicación web. Verifica que la aplicación funciona.

1. Servlet con Mensaje de Bienvenida

Crea un servlet de Java llamado `Welcome` que genere una página web con un mensaje de bienvenida cuando se acceda a este, no debe pertenecer a ningún paquete, nombre de igual modo la clase a utilizar. La página generada debe mostrar el mensaje *¡Bienvenido a nuestro parcial de servlets y JSP!*. Utiliza y adapta el siguiente código pero agrega unas pocas líneas de html con texto, es decir, código html con texto de relleno (son libres de agregar lo que quieran o pueden agregar un archivo css).

```
// arhivo servlet
import java.io.*;
import jakarta.servlet.*;
import jakarta.servlet.http.*;
import jakarta.servlet.annotation.*;

public class ClassName extends HttpServlet {

   @Override
   public void doGet(HttpServletRequest request, HttpServletResponse
response)
         throws IOException, ServletException {

      response.setContentType("text/html; charset=UTF-8");
      PrintWriter out = response.getWriter();

      out.println("""
            <!DOCTYPE html>
            <html>
            <head><title>Servlet Segundo Parcial 2</title></head>
            <body>
              <h1>Agrega texto aqui!</h1>
              <p>Request URI: %s</p>
              <p>Protocol: %s</p>
              <p>Remote Address: %s</p>
              <p>A Random Number: <strong>%f</strong></p>
            </body>
            </html>
            """.formatted(request.getRequestURI(), request.getProtocol(),
                        request.getRemoteAddr(), Math.random()));
      out.close();

      System.out.println("hello world, to Tomcat!");   // Check Tomcat's
console for this message
   }
```

Crea el servlet y el servlet-mapping en el archivo de configuración `web.xml`. El servlet debe responder al patrón de URL `/bienvenido`, utiliza y adapta el siguiente código

```
<!-- ejemplo de implementación de servlet -->
  <servlet>
    <servlet-name>ServletName</servlet-name>
    <servlet-class>ServletClassName</servlet-class>
  </servlet>
```

```
  <servlet-mapping>
    <servlet-name>ServletName</servlet-name>
    <url-pattern>/url</url-pattern>
  </servlet-mapping>
```

2. Servlet de Descargar de Archivos

Ahora agrega un servelet que se encargue de descargar el archivo index.html, debes reparar e implementar el siguiente servelet. Este servlet debe responder al URL `descargar` y el servlet se debe llamar `Descargar` y estar dentro del paquete `etitc2`.

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;

public class Download extends HttpServlet {
        public void doGet(HttpServletRequest request, HttpServletResponse
response) throws IOException, ServletException {
                response.setContentType("application/html");
                ServletContext ctx = getServletContext();
                InputStream is = ctx.getResourceAsStream("/index.html");
                // Make sure to show the download dialog
        response.setHeader("Content-disposition","attachment;
filename=index.html");
                int read = 0;
                byte[] bytes = new byte[5120];
                OutputStream os = response.getOutputStream();
                while ((read = is.read(bytes)) != -1) {
                        os.write(bytes, 0, read);
                }
                os.flush();
                os.close();
        }
}
```

Compila y agrega este servlet al arhivo de configuración.

Además, crea un archivo llamado `descarga.html` en la raiz de la aplicación que llame el servlet usando el método GET y la acción `descargar`. Utiliza el siguiente código, no es necesario modificarlo:

```
<html>
<head>
 <meta charset="utf-8">
</head>
<body>
        <h1 align="center">Descargar index.html</h1>
        <form method="GET" action="descargar">
                <p>¿Quieres descargar el archivo index.html?</p><input
```

```
type="SUBMIT" value="¡Descargar!">
        </form>
</body>
</html>
```

Para probar el servlet ingresa al URL `localhost:8080/segundo_parcial/descarga.html` y verifica que funciona correctamente el formulario, que al dar click en el botón de descargar debe descargar el archivo index.html.

3. Archivo JSP

Ahora crea una página jsp llamada `AutoresLibros.jsp` y colocala en la carpeta raíz de la aplicación.

```
<%@page language="java" contentType="text/html" pageEncoding="UTF-8" %>
<!DOCTYPE HTML >
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>Autores</title>
</head>

<body>
  <h2>Selecciona un autor:</h2>
  <form method="get">
    <input type="checkbox" name="author" value="Tan Ah Teck">Tan
    <input type="checkbox" name="author" value="Mohd Ali">Ali
    <input type="submit" value="Query">
  </form>

  <%
  String[] authors = request.getParameterValues("author");
  if (authors != null) {
  %>
    <h3>Has selecionado el(los) autore(s) :</h3>
    <ul>
      <%
      for (String author : authors) {
      %>
        <li><%= author %></li>
      <%
      }
      %>
    </ul>
  <%
  }
  %>
  <br /><a href="<%= request.getRequestURI() %>">Atras</a>
<body>
</html>
```

A este archivo agrega las siguientes modificaciones:

2. Primero crea un encabezado con un número aleatorio (clase java.util.Random).

3. Agrega un encabezado h2 con la fecha actual (clase java.util.Date) y otro con el día, (método getDays() de la clase Date).

4. Modifica el código para que se utilice un ciclo for para definir los posibles autores utilizando las listas:

    - autores: contiene los elementos que deben aparecer en el html.
    - apodos: contiene los valores que debe tener cada nombre

```
// código de llava para la creación de las listas
String[] autores = {"Francisco", "Lola", "Alejandra", "Paulo",
"Samanta"};
String[] apodos = {"pacho", "lola", "aleja", "luis", "paulo"};
```

    Para esto utiliza las directivas de jsp y recuerda que esta modificación se hace dentor del form.

4. Llamar un Servlet desde JSP

Ahora crea un archivo jsp llamado `dado.jsp` en la carpeta raíz de la aplicación que llame el servlet `Dice`. Utiliza los códigos que estan en el exámen y el siguiente código para que este jsp llame al servlet `Dice`, además debes cambiar el color de fondo.

```
<html>
<body bgcolor="color">

<!-- Forward to a servlet -->
<jsp:forward page="/url_servlet" />
</html>
```

Para llamarlo tienen que ir al URL `localhost:8080/segundo_parcial/dado.jsp`. El servlet a utilizar es `Dice.java` con dirección `\dice`, **este servlet es necesario compilarlo pero no deben agregarlo al web.xml**. Copien y peguen el siguiente servlet dentro de src.

```
// servlet Dice.java -- no modificar
import jakarta.servlet.*;
import jakarta.servlet.http.*;
import jakarta.servlet.annotation.*;
import java.io.*;

@WebServlet("/dice")
public class Dices extends HttpServlet {
        public void doGet(HttpServletRequest request, HttpServletResponse
response) throws IOException {
                PrintWriter out = response.getWriter();
                String d1 = Integer.toString((int) ((Math.random() * 6) +
```

```
1));
                String d2 = Integer.toString((int) ((Math.random() * 6) +
1));
                String d3 = Integer.toString((int) ((Math.random() * 6) +
1));

                out.println("<html> <body>"+
                "<h1 align=center>Dados al azar</h1>"+
                "<p>Resultado dado 1: " + d1 + "</p>"+
                "<p>Resultado dado 2: " + d2 + "</p>"+
                "<p>Resultado dado 3: " + d3 + "</p>"+
                "</body> </html>");
        }
}
```

5. Empaquetamiento de la Aplicación

Finalmente empaqueta la aplicación en un archivo war y verifica que este funciona, despliegalos con el nombre `segundo_parcial_2`. Para ello ve a la raíz de la aplicación y ejecuta el siguiente comando.

```
jar -cvf segundo_parcial_2.war *
```

Entregables

Al verificar que cada paso esta funcionando correctamente, cuando se ingresa al URL generado, tomen un pantallazo y agreguenlos a un archivo word, es decir, en cada paso tomen un pantallazo cuando este funcionan la implementación que hicieron y agreguenlo a un archivo word.

1. Escribe los archivos Java servlet para cada tarea.
2. Escribe los archivos JSP para cada tarea.
3. Compila los archivos servlet los cuales generarán archivos tipo class.
4. Archivo `web.xml` con las asignaciones de los servlets.
5. Crea un archivo war que contenga todos los archivos fuente Java, archivos JSP, los archivos servlet compilados y el `web.xml` actualizado.
6. Archivo word con los pantallazos del funcionamiento de cada paso.