PRÁCTICA 2 TOMCAT

Como ya sabemos, para crear una aplicación web que contenga funcionalidades en JAVA, necesitamos:

Archivo index.html: Este es el punto de entrada para la interfaz de usuario de tu aplicación web. Contiene el código HTML que se muestra en el navegador del usuario. Puede incluir referencias a hojas de estilo CSS, scripts JavaScript, y otros recursos necesarios para la interfaz de usuario.

Clase Servlet en Java: Los servlets son clases Java que manejan las solicitudes y respuestas en una aplicación web. Un servlet puede procesar datos, interactuar con bases de datos, realizar lógica de negocios y luego enviar una respuesta, que puede ser una página HTML, datos JSON, XML, etc. Cada servlet extiende la clase HttpServlet y sobrescribe métodos como doGet() o doPost() para manejar solicitudes GET o POST, respectivamente.

Archivo Descriptor de Despliegue (.xml): Este es un archivo de configuración XML utilizado en aplicaciones Java EE. Es conocido como "web.xml" en muchas aplicaciones web basadas en servlets. Define cómo se deben manejar las solicitudes, especificando detalles como qué URL están mapeadas a qué servlets, parámetros de inicialización del servlet, configuraciones de seguridad, entre otros.

Es importante destacar que esta es una arquitectura tradicional de aplicaciones web en Java. Las tecnologías y enfoques pueden variar, especialmente con la aparición de frameworks y plataformas más modernas como Spring Boot, que simplifican mucho la configuración y el despliegue de aplicaciones web en Java, pero para esta práctica utilizaremos dicha estructura.

En esta práctica, se pretende crear una aplicación web en Eclipse, utilizando el código adjunto más adelante. Se pide:

1. Crear 2 Workspaces en Eclipse, llamados Tomcat9 y Tomcat10.

2. Crear una Aplicación Web Dinámica en cada uno de ellos, y crear los archivos correspondientes con los códigos adjuntos en este documento. Hacerlo en ambos Workspaces.

3. En el Workspace Tomcat9, utilizaremos el servidor Tomcat embebido versión 9.0.84 para desplegar nuestra aplicación web creada, y en Workspace Tomcat 10, usaremos el servidor Tomcat embebido con versión 10.1.17, o similares.

4. ¿Qué problemas nos encontramos? ¿A qué se deben? ¿Cómo solucionarlos?

5. Desplegar nuestra aplicación web directamente con el servidor embebido desde eclipse, y creando un archivo .war y cargándolo al manager del servidor.

6. Actividad de **ampliación**: Crear nuestra propia aplicación web con total libertad temática y funcional, con los correspondientes archivos index, xml y servlets, y desplegarla en el servidor Tomcat.

index.html

```
<!DOCTYPE html>
<html>
<head>
	<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
crossorigin="anonymous"></script>

	<meta charset="utf-8">
	<meta name="viewport" content="width=device-width, initial-scale=1">
	<title></title>
</head>
<body>
	<div class="container mb-4">
		<div class="row">
			<form action="MediaServlet" method="post">
		<fieldset>
			<legend>Calculo media</legend>
			Numero 1: <input type="text" name="numero1" class="form-
control"><br>
			Numero 2: <input type="text" name="numero2" class="form-
control"><br>
			Numero 3: <input type="text" name="numero3" class="form-
control"><br>
		</fieldset>
		<input type="submit" value="Submit" class="btn btn-primary">
	</form>
	</div>
  </div>
</body>
</html>
```

MediaServlet.java

```
/*
* Para poder controlar el formulario tienes que crear una clase que herede (extends)
 * de HttpServlet
* con esta clase recuperas valores del formulario, los almacenas en variables, haces el
 * calculo y finalmente pintas un nuevo resultado.
 * Para poder hacer que funcione los métodos del servlet debes registrarlo. ¿Donde? en el
fichero web.xml.
* */

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class MediaServlet extends HttpServlet {

	/**
	 *
	 */
	private static final long serialVersionUID = 1L;

	/**
	 * ¿Por qué doPost y no doGet? Porque tu formulario (en el HTML es tipo post) -->
method="post">
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
	// Recoger los números
	double num1 = Double.parseDouble(request.getParameter("numero1"));
	double num2 = Double.parseDouble(request.getParameter("numero2"));
	double num3 = Double.parseDouble(request.getParameter("numero3"));

	// Calcular la media
	double media = (num1 + num2 + num3) / 3;

	// Preparar la respuesta
	response.setContentType("text/html");
	PrintWriter out = response.getWriter();
	out.println("<h1>La media es " + media + "</h1>");
	out.println("<h2>¡Voy a aprobar Despliegue sin problemas!</h2>");
	    }

}
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
 <display-name>AppWeb</display-name>

 <!--
         welcome-file-list Indica cual es el primer fichero que mostrar la aplicación
 -->
 <welcome-file-list>
  <welcome-file>index.html</welcome-file>
  <welcome-file>index.jsp</welcome-file>
  <welcome-file>index.htm</welcome-file>
  <welcome-file>default.html</welcome-file>
  <welcome-file>default.jsp</welcome-file>
  <welcome-file>default.htm</welcome-file>
 </welcome-file-list>


 <servlet> <!-- Indica que vas a registrar un servlet, es decir, una acción -->
  <servlet-name>MediaServlet</servlet-name> <!--  Indica como vas a llamar al servlet y lo
indicas en este caso en el action del form de html form action="MediaServlet" -->
  <servlet-class>MediaServlet</servlet-class> <!-- Indicas que clase java trata el
comportamiento -->
 </servlet>
 <servlet-mapping> <!-- Hace el mapeo del servlet, indicas el nombre de antes y lo asocias con
una url -->
  <servlet-name>MediaServlet</servlet-name>
  <url-pattern>/MediaServlet</url-pattern>
 </servlet-mapping>

</web-app>
```

El trabajo deberá incluir explicaciones, captura de pantalla, código realizado y cualquier recurso que consideres importante.

El documento se entregará con formato .pdf, y con nomenclatura UT5.2_Nombre_Apellidos, en la tarea correspondiente del aula virtual de la asignatura, y antes de que finalice el plazo establecido.