INAP
INSTITUTO NACIONAL DE ADMINISTRACIÓN PÚBLICA

Curso online: **Instalación, Configuración y Administración de Apache + Tomcat**

Módulo 3. Apache Tomcat

Capítulo 5. Administración de Apache Tomcat – Actividad 7 - Prácticas

Autores

Janine García Morera

Alexandra López de la Oliva Portugués

Julio Villena Román

Octubre de 2014

PRÁCTICAS

5.1. Cambio de puerto

Cambia de puerto el servidor Tomcat al 8880. Comprueba que los ejemplos siguen funcionando.

5.2. Usuario con permisos de administración

Añade un nuevo usuario con permisos de administración, es decir, que pueda acceder a la herramienta de administración.

5.3. Dos instancias simultáneas

Anteriormente quizás no sabías cómo arrancar más de una instancia de Tomcat simultáneamente en la misma máquina, porque daba error, pero ahora sí sabes hacerlo.

Arranca dos servidores Tomcat en la misma máquina, uno en el puerto 8080 y otro en el 8081.

5.4. Mi primer servlet

Antes hiciste tu primer JSP. Ahora el objetivo de este ejercicio es aprender a compilar, desplegar y ejecutar un servlet sencillo. Para desarrollar este ejercicio es necesario disponer del JDK de Java, igual que antes.

El primer paso es escribir el código del servlet `HolaMundo.java`, copiándolo del cuadro a continuación.

```
public class HolaMundo extends HttpServlet {
    /**
     * Método del servlet que responde a una petición GET.
     */
    public void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws IOException, ServletException
    {
        // establece ContentType y codificación de caracteres
        response.setContentType("text/html; charset=UTF-8");

        // obtiene un PrintWriter para escribir la respuesta
        PrintWriter out = response.getWriter();
```

```
        // escribe un documento HTML
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>¡Hola Mundo!</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>¡Hola Mundo!</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

Para compilar este servlet necesitas `HTTPServlet` y otras clases e interfaces de la API de Servlet. Por tanto, debes compilar con la opción `-cp`:

```
javac -cp ${CATALINA_HOME}/lib/servlet-api.jar:. HolaMundo.java
```

Quizás también necesites importar las clases necesarias añadiendo las siguientes líneas al principio del código del servlet:

```
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;
```

Desplegar un servlet consiste en incluir una serie de ficheros en un contenedor web (en nuestro caso, Tomcat) para que los clientes puedan acceder a su funcionalidad. Normalmente, el desarrollo de un servlet forma parte de lo que se denomina una aplicación Web, que no es más que una colección de servlets, páginas HTML, JSP, clases y otros recursos que se pueden empaquetar y ejecutar en distintos contenedores web, de distintos proveedores, y que ofrecen una determinada funcionalidad a la que los clientes acceden típicamente a través de un navegador.

Las aplicaciones Web, a partir de la especificación de Servlet 2.2, deben estructurarse según la siguiente jerarquía de subdirectorios:

Directorio raíz: contiene ficheros estáticos (HTML, imágenes, hojas de estilo, etc.) y JSPs.

- Directorio `WEB-INF`: debe contener un fichero `web.xml`. Este fichero configura la aplicación. Por ejemplo, permite declarar servlets, asignarles la URL de acceso y parámetros de inicio, declarar alias y filtros, etc.
- Directorio `classes`: debe contener los ficheros compilados (servlets, beans, etc.) de las clases utilizadas por la aplicación web.

- Directorio `lib`: debe contener otras bibliotecas de clases adicionales (comprimidas con `jar`) que utilice tu aplicación.
- Resto de subdirectorios: para ficheros estáticos y JSP.

Para desplegar una aplicación Web que contenga el servlet de ejemplo:

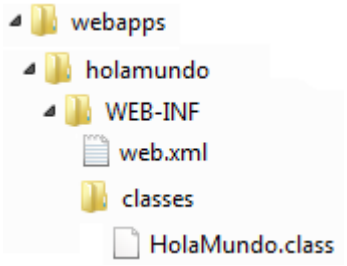
1. Crea un contexto para la aplicación web, denominado `holamundo`. Crear un contexto en Tomcat no es más que crear un directorio en `${CATALINA_HOME}/webapps/`. Este directorio será el ***directorio raíz*** de la aplicación web y, por lo tanto, debajo de él se debe crear la estructura de subdirectorios indicada anteriormente.

   El nombre del contexto es el primer nivel de la jerarquía de la ruta de acceso a la aplicación. Por tanto, en este caso la URL principal de la aplicación será: `http://localhost:8080/holamundo/`.

2. Copia el fichero de despliegue `web.xml`, como el cuadro.

```
<web-app>
  <servlet>
    <servlet-name>hola</servlet-name>
    <description>Servlet que presenta el mensaje "¡Hola
Mundo!".</description>
    <servlet-class>HolaMundo</servlet-class>
  </servlet>
  <servlet-mapping><servlet-name>hola</servlet-name><url-
pattern>/hola</url-pattern></servlet-mapping>
</web-app>
```

3. Sitúa correctamente los ficheros en la estructura de directorios creada:



4. Detén el servidor y vuelve a arrancarlo para que Tomcat cargue la nueva aplicación web.

Abre un navegador y conéctate a la URL `http://localhost:8080/holamundo/hola`. Esto hará que se ejecute el servlet de ejemplo, que devolverá la página con el mensaje ***¡Hola Mundo!***

La parte final de la URL del servlet es `/hola` porque se ha configurado así en el fichero `web.xml`. Observa el contenido de este fichero para entender cómo se configura un servlet y la URL a la que responde.

Haz un post en el foro de actividades con tus impresiones. Comenta cualquier aspecto que consideres conveniente relacionado con el ejercicio, así como cualquier problema o dificultad que hayas encontrado.