Ejercicios de desarrollo y configuración de aplicaciones web

Índice

1. Instalar una aplicación web más compleja

Vamos a crear un aplicación web más compleja con elementos dinámicos (*servlets*), que deberán ser compilados previamente. En este caso no trabajaremos directamente en el directorio de Tomcat, sino que trabajaremos en un directorio de desarrollo (nuestro proyecto Eclipse) y una vez construida la aplicación la desplegaremos en Tomcat.

1. Consideraciones previas

- **Directorios**: Dentro del proyecto Eclipse, estructuraremos los elementos de nuestra aplicación web en las siguientes carpetas
  - `src`: será el directorio de fuentes (ficheros .java, como los servlets que utilizaremos)
  - `web`: será el directorio para ficheros estáticos (páginas HTML estáticas, y fichero `web.xml` (dentro de su carpeta `WEB-INF`). Esta carpeta simulará la raíz de la aplicación web, a falta de los fuentes, que están aparte
  - `build`: será el directorio donde juntar toda la aplicación. Las clases compiladas deberán generarse en `build/WEB-INF/classes/`, el descriptor de despliegue lo colocaremos en `build/WEB-INF/web.xml`, las librerías (si las hay) en `build/WEB-INF/lib/` y las páginas (HTML o JSP) a partir de la raíz, en la carpeta o subcarpeta que queramos.
  - `dist`: será el directorio donde almacenar la aplicación empaquetada en un fichero WAR.
- **Classpath:**: Para poder compilar los fuentes, hemos de tener en cuenta que son servlets. Por lo tanto, además de la librería de clases de J2SE, deberemos añadir la librería `${tomcat.home}/common/lib/servlet-api.jar` al classpath, para poder trabajar con *servlets*.

2. Copia de ficheros

Copiar los ficheros que vienen sueltos en la plantilla a la carpeta que corresponda:

- Copiar el servlet `Conversor` a la carpeta `src` (creando el paquete que corresponda)
- Copiar los recursos estáticos de la web (ficheros `index.htm` y `web.xml`) en la carpeta `web`. El fichero `web.xml` copiadlo en una subcarpeta `WEB-INF` dentro de `web`

3. Fichero de ant

Editar el fichero `build.xml` de la plantilla para hacer un archivo de *ant* con los siguientes objetivos para poder compilar y desplegar la aplicación:

- Un objetivo `compile` que nos permita construir la aplicación. Este objetivo deberá:

  - Copiar los recursos estáticos (directorio `web`) al directorio `build`
  - Compilar el código fuente de `src` guardándolo en `build/WEB-INF/classes`
- Un objetivo `deploy` que realice el despliegue de la aplicación, copiando el contenido del directorio `build` al directorio de nuestra aplicación web de Tomcat. Por ejemplo, si llamamos `aplic` a nuestra aplicación, copiaremos todo su contenido al directorio `${tomcat.home}/webapps/aplic`. Iniciar Tomcat y comprobar que la aplicación funciona correctamente, accediendo a la siguiente URL:

  ```
  http://localhost:8080/aplic/index.htm
  ```

  Si hemos llamado `aplic` al contexto.
- Añadir un objetivo `dist` que cree el directorio `dist` y genere en él un fichero WAR con la aplicación web. Por ejemplo, `aplic.war`.

  Utilizar la interfaz HTML del *manager* de Tomcat para desplegar el fichero WAR creado en Tomcat. NOTA: Antes de desplegarlo deberemos eliminar la aplicación si la hemos desplegado anteriormente. Podemos utilizar el mismo *manager* para eliminarla.

  Para poder hacer esto necesitaremos contar con un usuario con permisos de `manager`. Si no tenemos ningún usuario con estos permisos lo crearemos editando el fichero `${tomcat.home}/conf/tomcat-users.xml` e introduciremos las siguientes líneas:

  ```
  <role rolename="manager"/>
  <user username="admin" password="j2ee" roles="manager"/>
  ```

  Con esto ya podremos acceder al manager utilizando el nuevo usuario `admin` con password `j2ee`.
- Modificar el objetivo `deploy` de *ant*, para que en lugar de copiar el directorio `build` directamente, utilice la tarea `deploy` (que se encuentra entre las tareas de Tomcat en `catalina-ant.jar`) para desplegar la aplicación.
- (*)También puedes añadir más objetivos al fichero de *ant*:
  - `undeploy`: Desinstala la aplicación web utilizando las tareas de Tomcat.
  - `reload`: Recarga la aplicación web utilizando las tareas de Tomcat.
  - `start`: Pone en marcha la aplicación web utilizando las tareas de Tomcat.
  - `stop`: Detiene la ejecución de la aplicación web utilizando las tareas de Tomcat.
  - `list`: Muestra la lista de aplicaciones instaladas en Tomcat.
  - `clean`: Elimina los directorio `build` y `dist`.
  - `all`: Limpia y compila desde cero.
  - `javadoc`: Genera la documentación Javadoc de nuestras clases.

2. Configuración de aplicaciones web

Vamos a instalar y configurar la aplicación web `conversor.war` que tenéis en la plantilla, cambiándole el contexto de la aplicación. Recordamos que para hacer esto hay varias alternativas:

- Añadir a mano un contexto para la aplicación en el fichero `server.xml`. Esta opción implica un cambio en un fichero principal de Tomcat, e incluso para la versión 5.5 está DESACONSEJADO, por lo que no lo haremos.
- Añadir un fichero XML en la carpeta `conf/nombre_engine/nombre_host` del Host (en nuestro caso `conf/Catalina/localhost`). Esto es más recomendable
- Instalar la aplicación desde el manager de Tomcat

Probaremos la segunda forma, para lo cual seguiremos estos pasos:

- Desinstalar cualquier versión del conversor que tuvierais instalada previamente en Tomcat, borrándola del `webapps` (o arrancando Tomcat y quitándola con el manager)
- Copiad el fichero `conversor.war` a una carpeta temporal donde haremos las pruebas, por ejemplo, en `C:/temp`
- Añadir un fichero XML en la carpeta `conf/Catalina/localhost` (por ejemplo, `conversor.xml`), y rellenarlo con una etiqueta `Context` como:

  ```
  <Context docBase="C:/temp/conversor.war" path="/util/conversor">
  </Context>
  ```

  Sustituid el `docBase` por la ruta donde hayáis copiado el fichero WAR.

  Fijaos también que cambiamos el `path` para llamar a la aplicación con una ruta diferente, en lugar de:

  ```
  http://localhost:8080/conversor
  ```

  la llamaremos con:

  ```
  http://localhost:8080/util/conversor
  ```
- Arrancad Tomcat si no lo teníais en marcha, y probad a acceder a esta dirección
- Copiad el fichero XML de configuración del contexto en vuestro proyecto Eclipse, como entrega de este ejercicio.

IMPORTANTE: es posible que para que reconozca el nuevo path `util/conversor` tengáis que renombrar el fichero XML que habéis añadido a `conf/Catalina/localhost` para que se llame igual que el path, cambiando las barras / por almohadillas # (fichero `util#conversor.xml`)

(*)Cambiar el contexto **mediante el formulario del manager de Tomcat**, haciendo que funcione igual que en el caso anterior. Para que funcionen los cambios así, habrá que desactivar previamente el despliegue automático de aplicaciones (poniendo `autoDeploy` y `liveDeploy` a `false` en el fichero `server.xml`). Tener en cuenta que cuando el

formulario solicita la URL del fichero de configuración XML y la del WAR, habrá que poner una del tipo `file:C:/Archivos de programa/Apache Software Foundation/Tomcat 5.5/conf/Catalina/localhost/fichero.xml` y `file:C:/temp/fichero.war`, respectivamente.

3. Registro de accesos

- Definir un `Valve` para registrar únicamente los accesos a la aplicación `conversor`. Dichos accesos deben quedar registrados en el directorio **accesos** dentro del principal de Tomcat. El nombre del fichero de accesos debe comenzar por **"conversor"** y tener extensión **".log"**. Probar a hacer peticiones para ver cómo quedan registrados los accesos.
- (*)Definir un `Valve` para que no pueda acceder a esta aplicación vuestro/a compañero/a de al lado, sabiendo su IP. Comprobar que efectivamente no puede hacerlo.

Los `Valves` que se añadan se pondrán en el mismo fichero XML que se creó en el ejercicio anterior, dentro de la etiqueta `Context`. Actualizad después en el proyecto el fichero XML con los cambios añadidos.