

Ejercicios de herramientas para desarrollo de aplicaciones web

Índice

1 Desarrollo y despliegue con Eclipse Webtools.....	2
2 Desarrollo y despliegue con Ant.....	2

1. Desarrollo y despliegue con Eclipse Webtools

Vamos a tomar la aplicación de "comentarios al sitio web" que desplegamos manualmente en la sesión anterior y prepararla para trabajar con Eclipse y Ant.

1. **Definir el servidor web en Eclipse.** Lo primero será definir un servidor Tomcat 6.0 en Eclipse, siguiendo los pasos que aparecen en los apuntes de la sesión.
2. **Crear el proyecto:** crearemos un *dynamic web project*, con el nombre `comentarios`. En el asistente para crearlo, pondremos como *Context Root* (paso 3 del asistente), **comentarios**, y dejaremos que nos cree las carpetas por defecto de `src` y `WebContent`.

Una vez lo tenemos creado, vamos a ir copiando en los lugares respectivos los siguientes ficheros que tenemos en la plantilla:

- Crea un paquete java llamado `ua` en la carpeta `src`
 - Copia en ese paquete los ficheros Java `Comentario` y `addComentario` de la plantilla
 - Copia las páginas web `comentarios.jsp` y `comentarioOK.html` en `WebContent`
 - Copia el descriptor de despliegue (`web.xml`) en la carpeta `WebContent/WEB-INF`
3. **Probar el proyecto:** Una vez lo tengas todo en su sitio y no haya errores, despliega la aplicación en el servidor de prueba. Para ello, ve con el botón derecho del ratón sobre el proyecto, y eliges *Run - Run As - Run on Server*. Aparecerá un mensaje de error, porque no existe página `index.jsp` ni `index.html`. Arreglaremos esto en la siguiente sesión. Por el momento, cambia manualmente la URL por `http://localhost:8080/comentarios/comentarios.jsp`.

2. Desarrollo y despliegue con Ant

Preparar la estructura de directorios: necesitaremos crear los directorios `build` y `dist` dentro del proyecto de Eclipse.

Crear el build.xml: 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 `WebContent`) 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*.

- 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.

