Servidores Web

Sesión 2: Desarrollo de aplicaciones web con Eclipse Webtools y Ant

Puntos a tratar

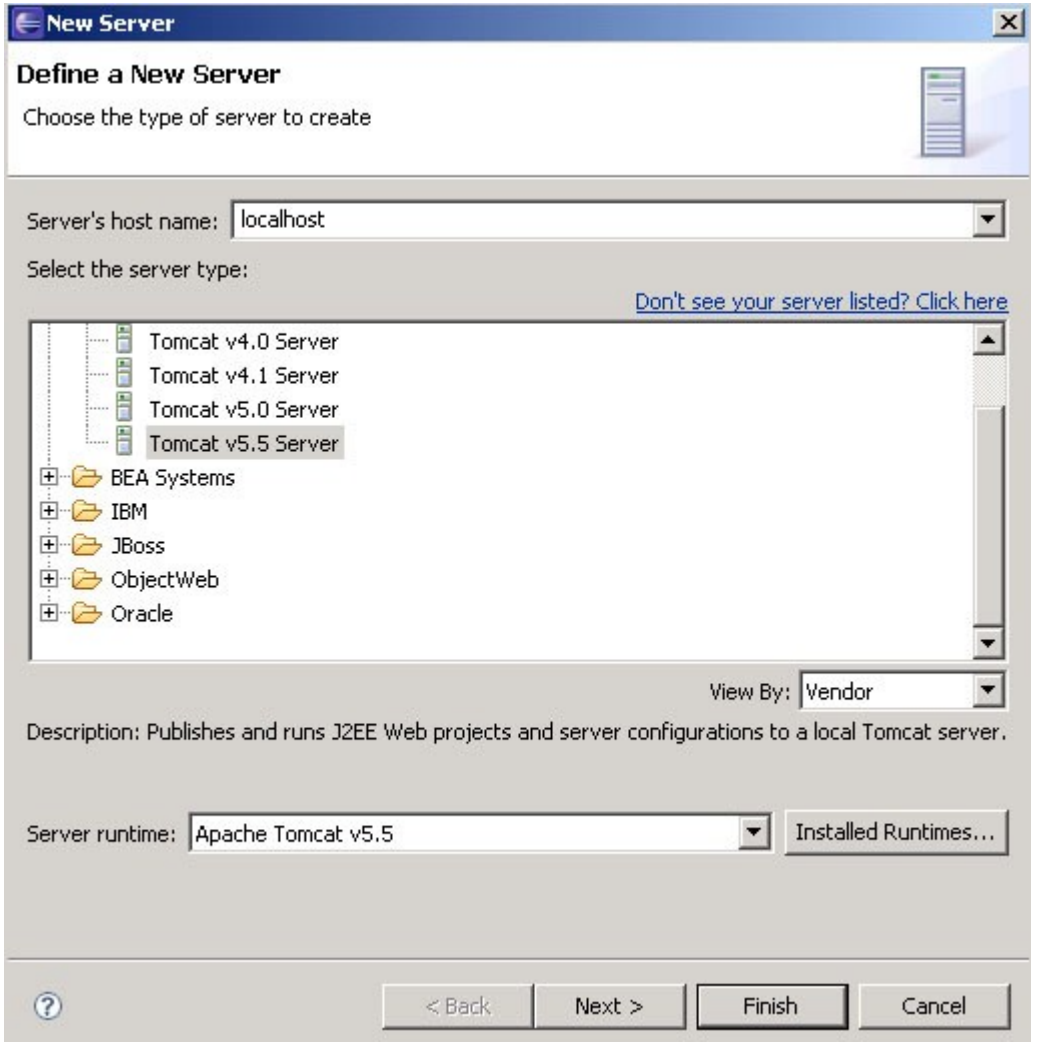
- Desarrollo de aplicaciones web con Webtools
  - Gestión del servidor web
  - Creación y desarrollo de la aplicación
  - Despliegue
- Desarrollo de aplicaciones web con Ant
  - Construcción de la aplicación
  - Despliegue
    - Exportando el .war
    - Tareas de Ant para despliegue

WebTools

- Es un conjunto de plugins de Eclipse que gestionan aplicaciones web como proyectos Eclipse
  - En *eclipse.org* se distribuye como “Eclipse para JavaEE” (actualmente *Eclipse Ganymede*)
- Podremos:
  - Gestionar el servidor web en que desplegar
  - Crear y desarrollar la aplicación web
  - Desplegar y probar la aplicación en el servidor

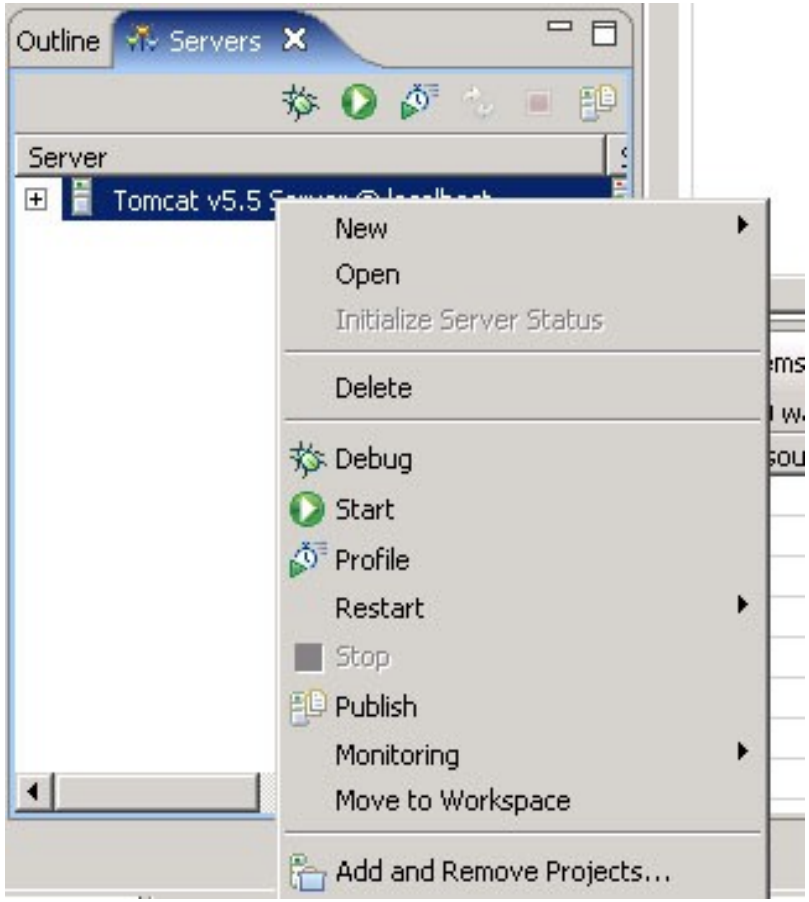
Añadir servidores web

- *File > New > Other > Server*



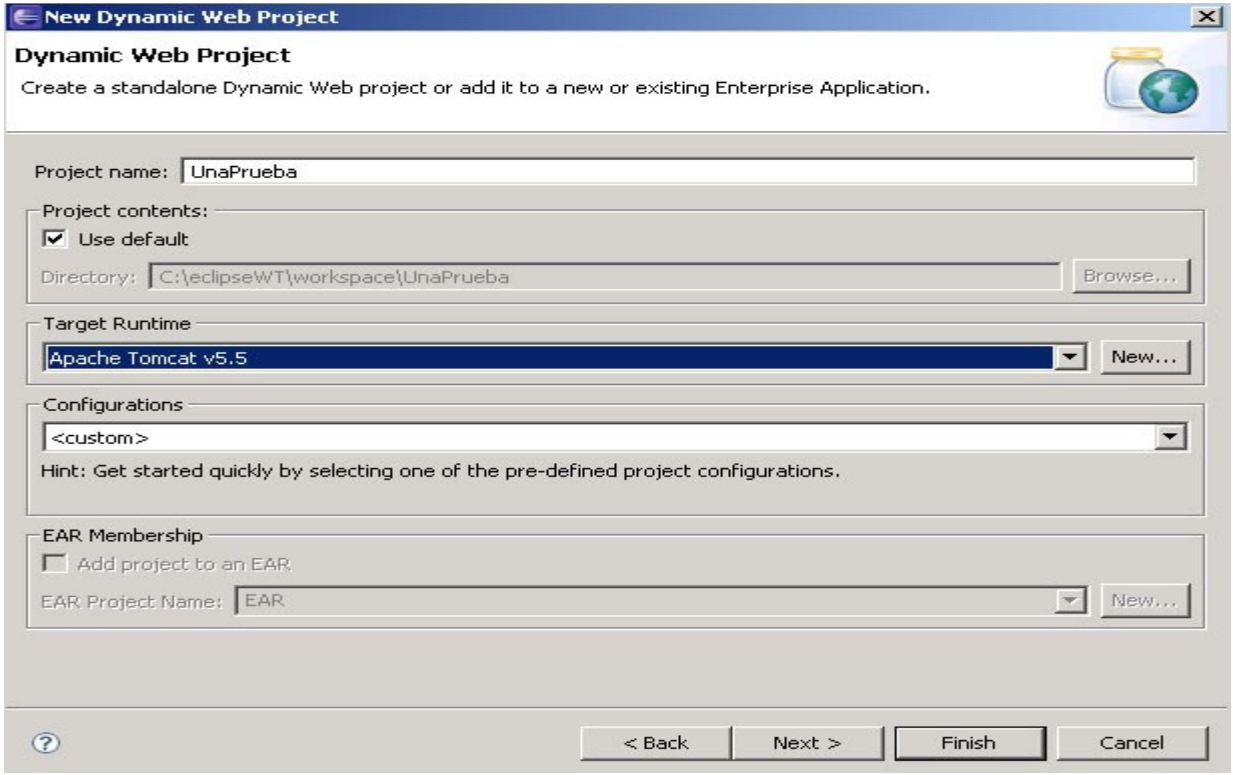
Gestionar los servidores web

- Botón derecho sobre el servidor en la solapa ***Servers***
- Tenemos opciones para pararlo, reanudarlo, etc
- La solapa ***Console*** muestra la salida del servidor



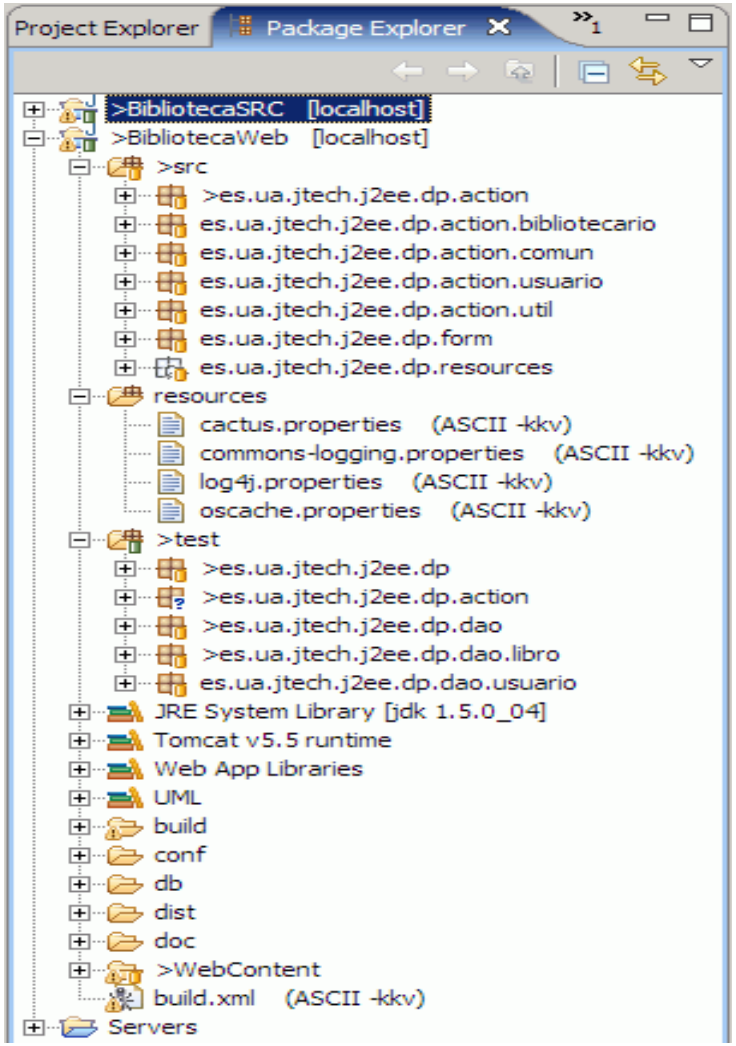
Crear proyecto de aplicación web

- Ir a *File >New > Project…* y elegir *Web – Dynamic Web Project*



Crear proyecto de aplicación web (2)

- En los siguientes pasos del asistente, elegimos qué carpetas crear y la ruta del contexto que tendrá la aplicación
- Carpetas creadas por defecto:
  - **src**: fuentes
  - **WebContent**: esqueleto aplicación web (con WEB-INF y sus subcarpetas)
  - El resto de carpetas las crearemos nosotros manualmente



Despliegue de la aplicación

- La aplicación se desplegará sobre el servidor que tengamos asignado en la vista *Servers*.
- Pulsamos botón derecho sobre el proyecto web y elegimos *Run As* > *Run on Server*
  - En la siguiente pantalla podemos elegir sobre qué servidor de la vista de *Servers* ejecutarlo, si tuviésemos más de uno configurado.
- Repetiremos la operación tras cada cambio que queramos comprobar en la aplicación.
- También se puede exportar como un .war en el directorio de despliegue del servidor (**webapps** en Tomcat) (*File* > *Export* > *WAR file*)

Desarrollo con Ant

- Útil cuando necesitamos realizar tareas complejas para las que no podemos configurar Eclipse
- Se suele usar una estructura de directorios estándar (no necesariamente coincide con la de Eclipse)
  - `src`: código fuente
  - `web o WebContent`: elementos que no necesitan ser compilados (recursos estáticos, JSPs, `WEB-INF/web.xml`, librerías)
  - `build`: Aplicación completa tal como se instalará en el servidor
  - `dist`: Aplicación empaquetada (fichero WAR)
  - *... veremos otros directorios relevantes más adelante*

Preparación

- Crear el directorio `build`
- Copiar el contenido de `web (o WebContent)` a `build`

```
<target name="prepare">
  <mkdir dir="${build.home}"/>
  <mkdir dir="${build.home}/WEB-INF"/>
  <mkdir dir="${build.home}/WEB-INF/classes"/>
  <mkdir dir="${build.home}/WEB-INF/lib"/>
  <copy todir="${build.home}">
    <fileset dir="${web.home}"/>
  </copy>
</target>
```

Compilación

- Compilar los fuentes de `src`
- Generar la salida en `build/WEB-INF/classes`

```
<target name="compile" depends="prepare"
        description="Compila los fuentes Java">
  <javac srcdir="${src.home}"
         destdir="${build.home}/WEB-INF/classes"
         debug="${compile.debug}"
         deprecation="${compile.deprecation}"
         optimize="${compile.optimize}">
    <classpath refid="compile.classpath"/>
  </javac>
</target>
```

Classpath

- En el *classpath* debemos tener las librerías de Java EE para compilar
- Cogemos la implementación de Java EE incluida en el servidor de aplicaciones que vamos a usar
  - En Tomcat se incluye parte de la especificación de Java EE

```
<path id="compile.classpath">
  <pathelement location="${catalina.home}/common/classes"/>
  <fileset dir="${catalina.home}/common/endorsed">
    <include name="*.jar"/>
  </fileset>
  <fileset dir="${catalina.home}/common/lib">
    <include name="*.jar"/>
  </fileset>
  <pathelement location="${catalina.home}/shared/classes"/>
  <fileset dir="${catalina.home}/shared/lib">
    <include name="*.jar"/>
  </fileset>
</path>
```

Empaquetar

- Opcional según el servidor
- Empaquetar el contenido de `build` en un fichero WAR, y guardarlo en `dist`
- El fichero WAR se crea con la herramienta JAR

```
<target name="dist" depends="compile"
        description="Crea el fichero WAR de la
                     aplicacion">
  <mkdir dir="${dist.home}"/>
  <jar jarfile="${dist.home}/${war.name}"
       basedir="${build.home}"/>
</target>
```

Directorio de aplicaciones

- El servidor web Tomcat tiene un directorio donde se encuentran las aplicaciones web instaladas

```
${tomcat.home}/webapps
```

- Cada aplicación web (o contexto) se guarda como un subdirectorio de `webapps`
- Al contexto se le da por defecto como ruta el nombre de este directorio
  - Si está en `webapps/aplic`, la ruta del contexto será

```
http://localhost:8080/aplic
```

  - El contexto `ROOT` creado por defecto tiene la ruta

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

  - Podremos cambiar estas rutas en la configuración

Despliegue

- Entendemos por *despliegue* (*deployment*) el proceso de instalar la aplicación web en el servidor de aplicaciones para que empiece a funcionar
- Alternativas para el despliegue:
  - Copiar la aplicación al directorio `webapps` de Tomcat. Copiamos el directorio de la aplicación o el fichero WAR
  - Utilizar la interfaz HTML del *manager* de Tomcat para subir el fichero WAR
  - Utilizar las tareas de *ant* para el despliegue:
    Añadir tareas de Tomcat a *ant*
    Utilizar la tarea de despliegue para subir el fichero WAR

Despliegue copiando la aplicación

- Para desplegar aplicaciones en Tomcat podemos copiarlas directamente al directorio `webapps`
  - Podemos copiar el directorio completo o el fichero WAR con la aplicación (Tomcat creará el directorio)

```
<target name="deploy" depends="compile"
        description="Despliega la aplicacion">
  <mkdir dir="${catalina.home}/${app.name}"/>
  <copy todir="${catalina.home}/${app.name}">
    <fileset dir="${build.home}"/>
  </copy>
</target>
```

Tareas de Ant

- Añadir librería `catalina-ant.jar` al `CLASSPATH` o a la configuración de Eclipse (*Window – Preferences – Ant – Runtime*)

```
${tomcat.home}/server/lib/catalina-ant.jar
```

- Declarar las tareas de Tomcat

```
<taskdef name="deploy"    classname="org.apache.catalina.ant.DeployTask"/>
<taskdef name="install"   classname="org.apache.catalina.ant.InstallTask"/>
<taskdef name="list"      classname="org.apache.catalina.ant.ListTask"/>
<taskdef name="reload"    classname="org.apache.catalina.ant.ReloadTask"/>
<taskdef name="remove"    classname="org.apache.catalina.ant.RemoveTask"/>
<taskdef name="resources" classname="org.apache.catalina.ant.ResourcesTask"/>
<taskdef name="roles"     classname="org.apache.catalina.ant.RolesTask"/>
<taskdef name="start"     classname="org.apache.catalina.ant.StartTask"/>
<taskdef name="stop"      classname="org.apache.catalina.ant.StopTask"/>
<taskdef name="undeploy"  classname="org.apache.catalina.ant.UndeployTask"/>
```

Despliegue con Ant

- Se puede automatizar el despliegue con la tarea `deploy`

```
<target name="deploy" depends="dist"
        description="Despliega la aplicacion">
  <deploy url="${manager.url}"
          username="${manager.nombre}"
          password="${manager.passwd}"
          path="${app.path}"
          war="file:${dist.home}/${war.name}"/>
</target>
```