Java EE – Apache Tomcat

Aplicaciones Web / Sistemas Web

Juan Pavón Mestras
Dep. Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense Madrid

Tomcat

- Apache Jakarta (http://jakarta.apache.org/)
  - Proyectos de herramientas Java (1999-2011)
  - Ant, Commons, Gump, JMeter, Lucene, Maven, Struts, Tomcat, ...
- Tomcat es un contenedor de servlets
  - Puede utilizarse como un servidor de aplicaciones Web con HTML, servlets y JSPs
  - O como complemento al servidor Apache
- Bien integrado en eclipse
- Implementación de referencia para Java Server Pages (JSP) y Java Server Faces (JSF)
- Página oficial: http://tomcat.apache.org/

Arquitectura

- Servidor HTTP
- Contenedor de servlets
  - Ejecuta servlets
  - Convierte páginas JSP y JSF en servlets
- Arquitectura jerárquica y modular:

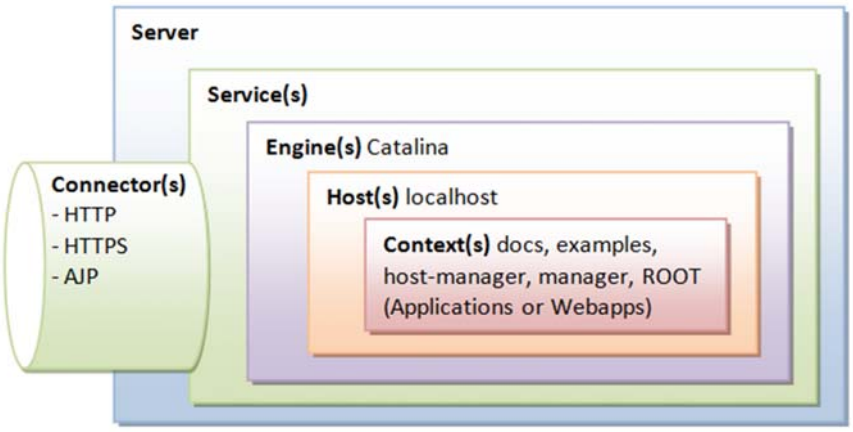


Figura de: http://www.ntu.edu.sg/home/ehchua/programming/howto/Tomcat_More.html

Organización de directorios

- bin – ejecutables y scripts de Tomcat
- conf – configuración global para todas las aplicaciones
  - catalina.policy – políticas de seguridad
  - catalina.properties y logging.properties
  - servlet.xml – fichero de configuración principal de Tomcat
  - web.xml – descriptores de despliegue de aplicación web
  - context.xml – opciones de configuración específicas de Tomcat
  - tomcat-users.xml – base de datos de usuarios y passwords
  - Un subdirectorio por cada engine y host. P.ej. Catalina/localhost
- lib – ficheros JAR disponibles para todas las aplicaciones web
  - servlet-api.jar (Servlet), jasper.jar (JSP) y jasper-el.jar (EL)
  - Drivers para bases de datos: MySQL JDBC driver (mysql-connector-java-5.1.{xx}-bin.jar)
  - JSTL (jstl.jar y standard.jar).
- logs – ficheros de logs
  - Catalina.{yyyy-mm-dd}.log, localhost.{yyyy-mm-dd}.log, etc.
- webapps – directorio base para las aplicaciones web
- work – servlets y clases resultantes de traducir ficheros JSP y JSF
- temp – ficheros temporales

Aplicaciones Web con Tomcat

- Las aplicaciones Web constan de varias partes que se organizan en varios directorios
  - Directorio raíz de la aplicación Web
    - Ficheros HTML, JSP, CSS, JS, imágenes, etc. que son visibles a los clientes de la aplicación
  - /WEB-INF/web.xml
    - Web Application Deployment Descriptor
      - Fichero XML que describe los servlets y otros componentes de la aplicación, además de parámetros de inicialización y restricciones de seguridad
  - /WEB-INF/classes/
    - Clases Java y recursos asociados: Servlets y no servlets que no estén contenidos en ficheros JAR
  - /WEB-INF/lib/
    - Ficheros JAR: Librerías de clases, drivers JDBC, etc.
- Además hay librerías compartidas en el contenedor
  - En $CATALINA_HOME/lib
    - APIs Servlet 3.0 y JSP 2.1
    - XML Parser conforme a JAXP para procesar documentos XML

web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
                version="2.4">

  <servlet>
      <servlet-name>Nombre del Servlet</servlet-name>
      <servlet-class>es.ucm.cursoweb.MiServlet</servlet-class>
      <init-param>
          <param-name>parametro</param-name>
          <param-value>valor</param-value>
      </init-param>
  </servlet>

  <servlet-mapping>
      <servlet-name>Nombre del Servlet</servlet-name>
      <url-pattern>/saluda</url-pattern>
  </servlet-mapping>
</web-app>
```

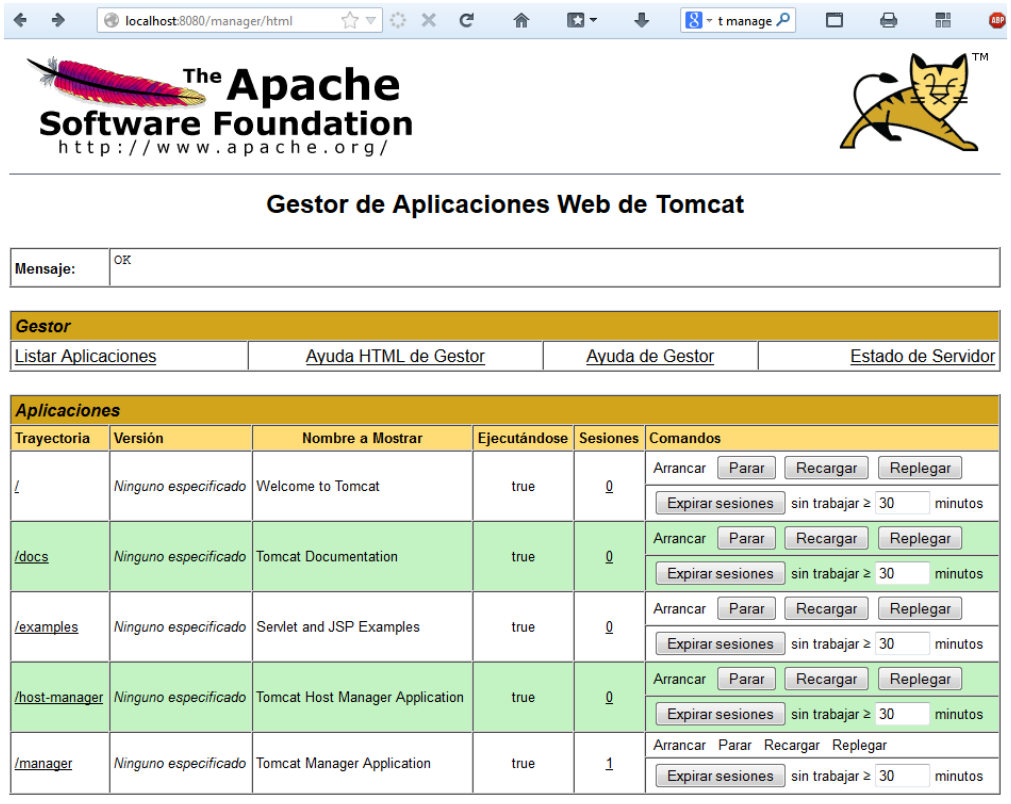
Gestión de Tomcat

- La conexión al gestor de Tomcat se hace en la dirección **http://localhost:8080/manager/html**
- Hay que definir los usuarios que pueden gestionar Tomcat
  - Los usuarios se definen con roles en el fichero <tomcat>/conf/tomcat-users.xml

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="tomcat"/>
  <role rolename="manager-gui"/>
  <user username="admin" password="passwd" roles="tomcat,manager-gui"/>
</tomcat-users>
```

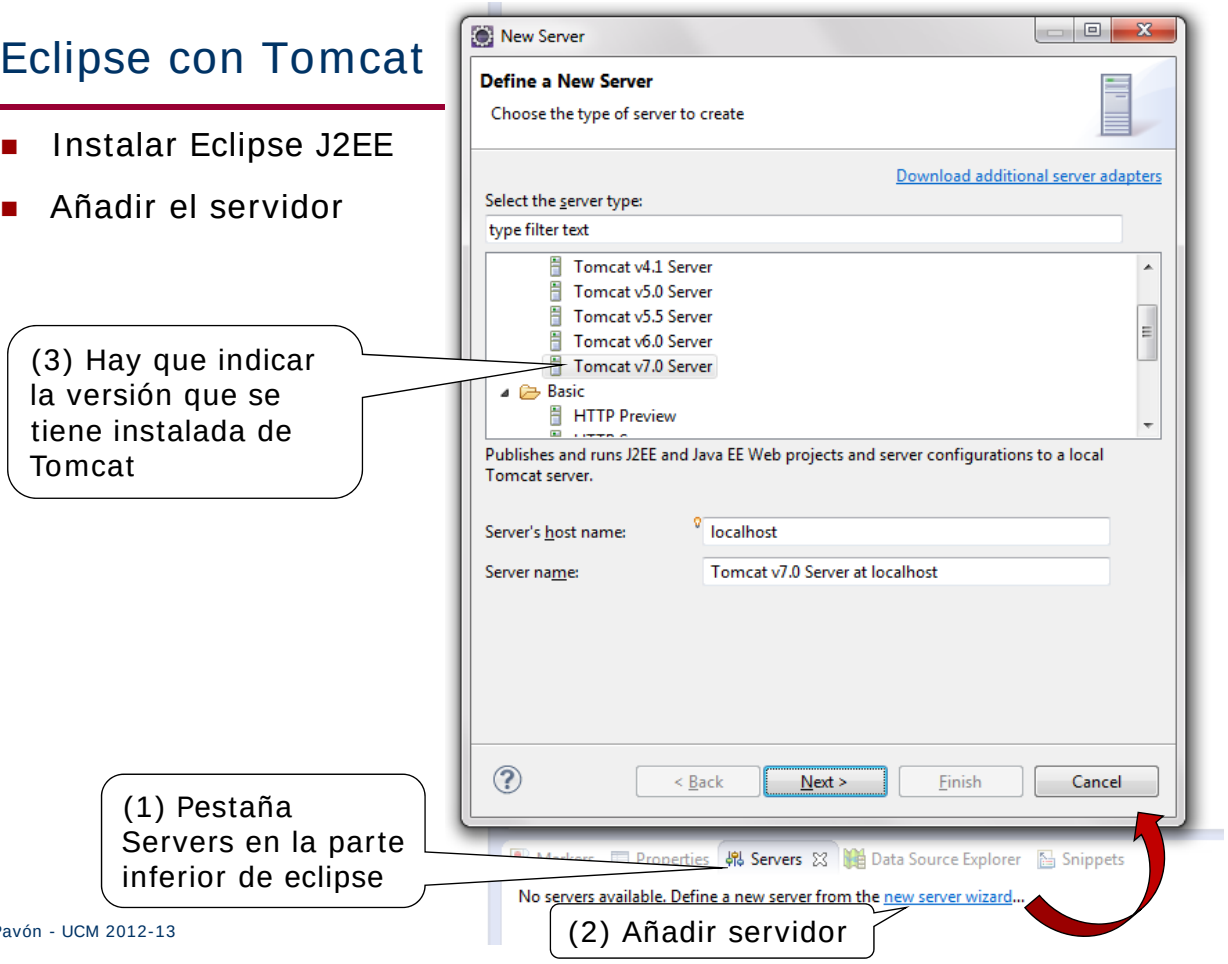
  - Para entrar habrá que especificar el usuario (admin) con su password

Gestor de Aplicaciones de Tomcat



Eclipse con Tomcat

- Instalar Eclipse J2EE
- Añadir el servidor



Eclipse con Tomcat

- Desde eclipse se puede arrancar y parar el servidor Tomcat seleccionando el servidor y con las opciones del menú contextual
- Para probarlo, intentar acceder a http://localhost
  - La primera vez lo normal es que salga un error 404
    - Hay que copiar el contenido del directorio ROOT de apache-tomcat-7.0.34\webapps en el correspondiente del workspace de eclipse: *workspace*\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps
- Un ejemplo para probar una primera aplicación:

http://courses.coreservlets.com/Course-Materials/servlet+jsp-code/test-app.zip

  - Descargarlo y luego desde eclipse probar File → Import → General → Existing Projects → Select archive file
  - Seleccionar el servidor Tomcat v7.0 y añadir test-app con "Add and Remove" en el menú contextual
  - Probarlo con http://localhost:8080/test-app/
    que ofrece enlaces a otras páginas del ejemplo

Adaptado de: *Tutorial: Installing Tomcat 7 and Using it with Eclipse*
http://www.coreservlets.com/Apache-Tomcat-Tutorial/tomcat-7-with-eclipse.html

Crear una aplicación J2EE con Eclipse

- Crear un nuevo proyecto
  - File → New → Project → Web → Dynamic Web Project
    - Para el "Target Runtime", elegir "Apache Tomcat v7.0"
    - Dar un nombre al proyecto
    - Aceptar las demás opciones por defecto y hacer Next dos veces hasta la última pantalla del wizard
    - Marcar entonces la casilla "Generate web.xml deployment descriptor"

Organización de los ficheros de la aplicación web

- Directorios de un proyecto de aplicación web en eclipse
  - WebContent
    - Ficheros típicos de web (HTML, JavaScript, CSS, JSP, imágenes, etc.)
      - Aquí es habitual tener un fichero index.html o index.jsp
    - Pueden organizarse en subdirectorios
  - WebContent/WEB-INF
    - web.xml – descriptor de despliegue
    - Se puede omitir en servlet 3.0 apps, si se hacen los mappings de servlet con las anotaciones @WebServlet en el código Java
  - WebContent/WEB-INF/lib
    - Ficheros JAR específicos de la aplicación
  - src/paquete
    - Código Java del paquete
    - Para crear un paquete hacer New package en "Java Resources: src"
      - Usar siempre paquetes. No es nada recomendable usar el default

Alternativas a Tomcat

- Apache TomEE
  - Adaptación de Tomcat para integrar la funcionalidad de J2EE 6:
    - Contexts and Dependency Injection (CDI)
    - EJBs
    - RESTful Web Service con JAX-RS
- Glassfish
  - Desarrollado por Sun (Oracle): https://glassfish.java.net/
- JBoss
  - Incluye EJBs, programación orientada a aspectos (AOP), servicio de persistencia de objetos (Hibernate), gestión de caché, servicios de mensajería, etc.
  - http://www.jboss.org
- WebLogic
  - Servidor Web J2EE de Oracle
  - http://www.oracle.com/es/products/middleware/appserver/
- IBM WebSphere Application Server (WAS)
  - http://www-03.ibm.com/software/products/es/es/appserv-was/

Bibliografía

- Instalación y configuración de Tomcat con Eclipse
  - http://www.coreservlets.com/Apache-Tomcat-Tutorial
- Descripción de Tomcat, arquitectura e implementación
  - http://www.ntu.edu.sg/home/ehchua/programming/howto/Tomcat_More.html
- Uso de eclipse y maven
  - Jesús L.C. *Configurar maven para hacer deploy en Tomcat*, en http://jesuslc.com/2013/04/08/358/
  - How to create a Maven web app and deploy to Tomcat – fast
    Added by Cody Burleson, last edited by Ben Shoemate on Jan 26, 2013

https://wiki.base22.com/display/btg/How+to+create+a+Maven+web+app+and+deploy+to+Tomcat+-+fast

Apéndice: Desarrollo con Tomcat y Maven

- Prerrequisitos
  - Maven está instalado
  - Tomcat está instalado y configurado para correr en el puerto 8080
  - Eclipse J2EE (opcional)

- Procedimiento
  1. Crear una nueva aplicación Web con Maven
  2. Definir el servidor Tomcat en la configuración de Maven
  3. Apuntar el Pom al servidor Tomcat
  4. Construir e implantar la aplicación Web

Adaptado de: *How to create a Maven web app and deploy to Tomcat – fast*
Added by Cody Burleson, last edited by Ben Shoemate on Jan 26, 2013

https://wiki.base22.com/display/btg/How+to+create+a+Maven+web+app+and+deploy+to+Tomcat+-+fast

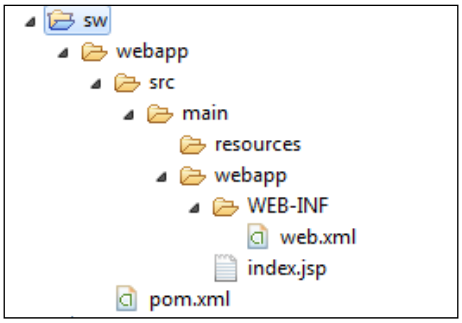
Apéndice: Desarrollo con Tomcat y Maven

1. Crear una nueva aplicación Web usando Maven

- Si se usa Eclipse, crear un proyecto general
  - New > Project... > General > Project
    - Project name: "sw" (o cualquier otro nombre que se desee)
  - Utilizando una consola, moverse al directorio del proyecto "sw" y ejecutar el siguiente comando de Maven (en una sola línea):

```
mvn archetype:create
      -DgroupId=es.ucm.sw
      -DartifactId=webapp
      -DarchetypeArtifactId=maven-archetype-webapp
```

- Esto creará la estructura del proyecto (para que Eclipse lo vea hacer Refresh)



Apéndice: Desarrollo con Tomcat y Maven

- El fichero **index.jsp** es el más elemental que se pueda imaginar:

```
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
```

- Y el de configuración de despliegue: **web.xml**

```
<!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
  <display-name>Archetype Created Web Application</display-name>
</web-app>
```


Apéndice: Desarrollo con Tomcat y Maven

2. Definir el servidor Tomcat en la configuración de Maven

- Editar el fichero settings.xml de Maven
  - Se puede ver en eclipse en Windows->preferences y una vez allí en Maven->user settings
    - La primera vez puede que no se encuentre el fichero settings.xml
    - Se creará entonces
    - Tras editarlo hacer **Update settings**
  - En el fichero, añadir el servidor "TomcatServer" con los credenciales para entrar como admin en el gestor de Tomcat

```
<settings>
    <servers>
    <server>
        <id>TomcatServer</id>
        <username>admin</username>
        <password>passwd</password>
    </server>
    </servers>
...
```

Apéndice: Desarrollo con Tomcat y Maven

3. Declarar en el Pom que se va a usar el servidor Tomcat

- Editar el fichero pom.xml del proyecto "sw" y reemplazar la sección <build> con lo siguiente:

```
<build>
    <finalName>sw</finalName>
    <plugins>
        <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>tomcat-maven-plugin</artifactId>
            <configuration>
                <url>http://127.0.0.1:8080/manager</url>
                <server>TomcatServer</server>
                <path>/sw</path>
            </configuration>
        </plugin>
    </plugins>
</build>
```


Apéndice: Desarrollo con Tomcat y Maven

4. Construir e implantar la aplicación Web

- En eclipse, seleccionar el pom.xml y hacer Run As-> Maven build e indicar **tomcat:deploy**
  - Alternativamente, desde la línea de comandos, ir al directorio sw/webapp donde está el pom y ejecutar

    mvn tomcat:deploy

- Si el resultado es BUILD SUCCESSFUL se puede acceder a la aplicación web en http://localhost:8080/sw/
  - Se tiene que ver el clásico "Hello World!" (de la página index.jsp)

- A partir de aquí ya se puede construir otra aplicación web
- NOTA: Si se intenta usar deploy otra vez dará FAIL porque la aplicación ya existe. Las siguientes veces habrá que hacer:

  mvn tomcat:redeploy