Sesión 5:
Introducción a Java EE

Índice

- Cliente-servidor y protocolo HTTP
- El servidor Tomcat
- Introducción a las aplicaciones web Java EE
- Servlets y JSP
- Visión global de Java EE

Introducción a Java EE

- **Cliente-servidor y protocolo HTTP**
- **El servidor Tomcat**
- **Introducción a las aplicaciones web Java EE**
- **Servlets y JSP**
- **Visión global de Java EE**

Cliente-servidor

- **Toda aplicación web se basa en el protocolo *cliente-servidor***
  - **El cliente (navegador) proporciona al usuario unas interfaces (formularios, enlaces, etc) para pedir páginas y acciones a un servidor**
  - **El servidor ejecuta las peticiones del cliente y muestra los resultados relacionados (listado de búsqueda, alta en un registro, etc)**

Protocolo HTTP

- **En este proceso interviene una serie de factores**
  - **La *petición* que realiza el usuario o cliente**
  - **Las *cabeceras de petición* que envía el cliente (normalmente añadidas automáticamente por el navegador), con información sobre la petición**
  - **La *respuesta* que le envía el servidor (p. ej, la página solicitada)**
  - **Las *cabeceras de respuesta* que envía el servidor, con información sobre la petición**
  - **Un *código de estado* enviado por el servidor, indicando si la petición se ha podido atender bien**
    - **p. ej, el código 404 indica un tipo de error**
- **Estos elementos se envían mediante el protocolo HTTP**

Introducción a Java EE

- Cliente-servidor y protocolo HTTP
- **El servidor Tomcat**
- Introducción a las aplicaciones web Java EE
- Servlets y JSP
- Visión global de Java EE

Tomcat

- **Tomcat es un servidor web que soporta gran parte de la arquitectura Java EE**
- **No es tan potente como un servidor de aplicaciones pero puede dar soporte a aplicaciones web bastante completas**
- **Más información en**

http://tomcat.apache.org/

Instalación

- **Ejecutable Windows: ejecutar el instalador y seguir los pasos**
- **Fichero ZIP (Windows o Linux): descomprimir en el lugar deseado**
- **Se deben definir dos variables (con el instalador no es necesario):**
  - **JAVA_HOME: con el directorio de instalación de Java**
  - **CATALINA_HOME: con el directorio de instalación de Tomcat**

Ejecución

- **Ejecutable Windows**: ***Inicio – Programas – Apache Tomcat 5.5 – Monitor Tomcat***
- **Fichero ZIP (Windows o Linux): ejecutables *startup* o *shutdown* de la subcarpeta *bin***
- **Una vez arrancado, abrir desde un navegador:**
  - ***http://localhost:8080***

Estructura

- Carpeta *bin*: ejecutables
- Carpeta *common*: clases y librerías compartidas
- Carpeta *conf*: configuración:
  - *server.xml*: fichero principal de configuración
  - *web.xml*: configuración general para aplicaciones web
  - *tomcat-users.xml*: lista de usuarios y contraseñas
  - *catalina.policy*: política de permisos
- Carpeta *webapps*: donde irán las aplicaciones web
- Carpeta *logs*: donde se volcarán los mensajes de log del servidor
- ... etc

Introducción a Java EE

- Cliente-servidor y protocolo HTTP
- El servidor Tomcat
- **Introducción a las aplicaciones web Java EE**
- Servlets y JSP
- Visión global de Java EE

Elementos a considerar

- **En el servidor:**
  - **La aplicación debe recoger parámetros y peticiones del cliente, procesarlos y devolver un resultado**
  - **Podrá valerse (de forma TRANSPARENTE para el cliente) de herramientas externas (servlets, JSP, PHP, ASP...)**
- **En el cliente:**
  - **HTML es estático y no permite más que dar formato y estructura a la información**
  - **Para cosas más complejas (validar formularios, mostrar evoluciones de datos, etc) podemos valernos de herramientas como Javascript, Flash, Applets, etc.**

Estructura de una aplicación Java EE

- **Directorio raíz /: desde él podemos colgar páginas HTML o JSP (estructuradas en subcarpetas si se quiere)**
- **Carpeta /*WEB-INF*: contiene la información web relevante para la aplicación:**
  - **Fichero descriptor */WEB-INF/web.xml***
  - **Librerías en */WEB-INF/lib/***
  - **Clases Java en */WEB-INF/classes/***
- **El resto de elementos (imágenes, etc) podemos ponerlos como queramos (aunque fuera de WEB-INF)**
- **CUALQUIER servidor web Java EE soporta esta estructura**
  - **Sólo hace falta copiar la carpeta con todo en el servidor correspondiente**

Contextos

- **Cada aplicación Java EE es un contexto, una unidad con sus recursos, clases y configuración**
- **Cada contexto o aplicación web tendrá asociada una ruta en el servidor web**
  - **Por ejemplo, para acceder a la aplicación *aplic* en nuestra máquina local:**
    - **http://localhost/aplic/index.html**
- **Podemos empaquetar la aplicación en un fichero WAR (con el comando *jar*) para que sea más fácil de distribuir o de llevar de un sitio a otro.**

Introducción a Java EE

- Cliente-servidor y protocolo HTTP
- El servidor Tomcat
- Introducción a las aplicaciones web Java EE
- Servlets y JSP
- Visión global de Java EE

Servlets

- **Un servlet es un programa Java que se ejecuta en un servidor web y construye o sirve páginas web**
- **Más sencillo de usar, eficiente, potente y portable que un CGI**
- **Para trabajar con servlets (y JSP) necesitamos:**
  - **Un servidor web que les dé soporte (p. ej. Tomcat)**
  - **Las librerías necesarias para trabajar con ellos (suelen venir en el servidor. En Tomcat son *servlet-api.jar* y *jsp-api.jar*)**
  - **Opcionalmente, la documentación sobre las clases**

Clases de servlets

- **Toda la arquitectura de servlets está en el paquete *javax.servlet* y derivados, de la librería de servlets**
  - **La interfaz *Servlet* define las características globales**
  - **La clase *GenericServlet* es una clase abstracta que implementa esa interfaz**
  - **La clase *HttpServlet* es un subtipo de la anterior, para servlets que procesen peticiones HTTP**
    - **Trabaja con elementos *ServletRequest (HttpServletRequest)* para recibir las peticiones de los clientes**
    - **Trabaja con elementos *ServletResponse (HttpServletResponse)* para enviar las respuestas a los clientes**

Estructura básica de un servlet


```
import javax.servlet.*;
import javax.servlet.http.*;
public class ClaseServlet extends HttpServlet
{
   public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
   throws ServletException, IOException
   {
       // ... codigo para una peticion GET
   }

   public void doPost(HttpServletRequest request,
                      HttpServletResponse response)
   throws ServletException, IOException
   {
       // ... codigo para una peticion POST
   }
}
```

Llamada de servlets

- **Mediante el alias o directorio virtual *servlet*, indicando la clase con sus paquetes:**

```
http://localhost:8080/miapp/servlet/paquete1.subpaquete1.MiServlet
```

- **Mapeando una ruta alternativa en *web.xml***

```
<servlet>
     <servlet-name>nombre</servlet-name>
     <servlet-class>paquete1.subpaquete1.MiServlet</servlet-class>
</servlet>
<servlet-mapping>
     <servlet-name>nombre</servlet-name>
     <url-pattern>/ejemploservlet</url-pattern>
</servlet-mapping>
```

```
http://localhost:8080/miapp/ejemploservlet
```

Ejemplos básicos (I)


```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class ClaseServlet extends HttpServlet
{
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
  throws ServletException, IOException
  {
      PrintWriter out = response.getWriter();
      out.println ("Este es un servlet de prueba");
  }
}
```

Ejemplos básicos (II)


```
...
public class ClaseServletHTML extends HttpServlet
{
   public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
   throws ServletException, IOException
   {
       response.setContentType("text/html");
       PrintWriter out = response.getWriter();
       out.println ("<!DOCTYPE HTML PUBLIC \""+
                "-//W3C//DTD HTML 4.0 " +
                "Transitional//EN\">");
       out.println ("<HTML>");
       out.println ("<BODY>");
       out.println ("<h1>Titulo</h1>");
       out.println ("<br>Servlet que genera HTML");
       out.println ("</BODY>");
       out.println ("</HTML>");
   }
}
```

Ejemplos básicos (III)


```
...
public class ClaseServletHTML2 extends HttpServlet
{
   public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
   throws ServletException, IOException
   {
       response.setContentType("text/html");
       response.sendRedirect("miPagina.jsp");
   }
}
```

Páginas JSP

- **Los servlets eran clases Java que generaban o mostraban contenido web**
- **Las páginas JSP son páginas HTML con código Java incrustado**
- **El *contenedor JSP* (una parte del servidor web) se encarga de sustituir el código Java por el resultado de su ejecución, y devolver la página HTML resultante al cliente**

Ejemplos de JSP

```
<html>
  <head>
      <title>Mi primera página JSP</title>
  </head>
  <body>
      <h1> Hoy es:
            <%= new java.util.Date() %>
      </h1>
  </body>
</html>
```

Scripts JSP


```
<%
   java.util.Calendar ahora =
   java.util.Calendar.getInstance();
   int hora =ahora.get(java.util.Calendar.HOUR_OF_DAY);
%>
<b> Hola mundo, <i>
<% if ((hora>20)||(hora<6)) { %>
  buenas noches
<% }
    else if ((hora>=6)&&(hora<=12)) { %>
        buenos días
<%       } else { %>
             buenas tardes
<%       } %>
```

```
Hoy es <%= new java.util.Date() %>
```

Otros objetos en JSP


```
<% reponse.sendRedirect("mipagina.jsp"); %>
```

```
<jsp:include page="otraPagina.jsp"/>
```

```
<jsp:forward page="error.html"/>
```

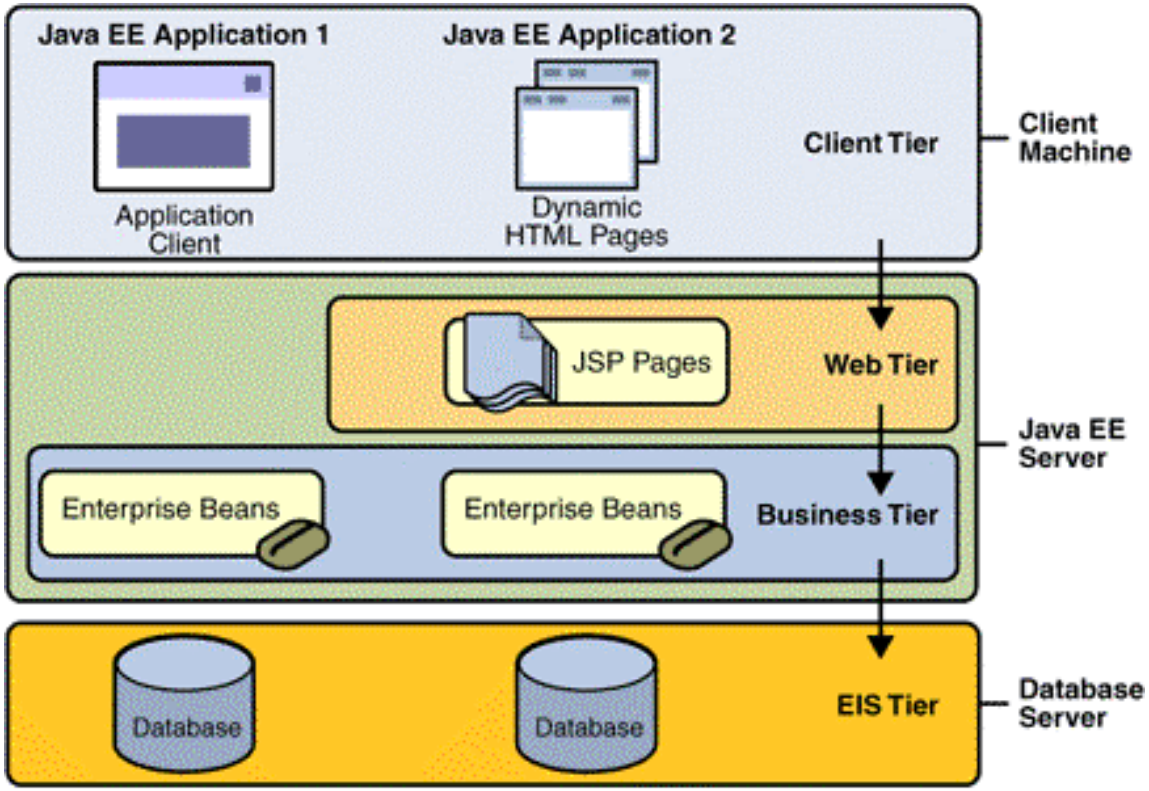
Relación servlets - JSP

- **El servidor web traduce internamente cada página JSP en un servlet**
  - **En principio servlets y JSP ofrecen la misma funcionalidad**
- **Sin embargo:**
  - **Los JSP son mejores para presentar la información (generar las páginas de contenidos)**
  - **Los servlets son mejores para procesar la información, y en función de la misma, decidir qué página JSP mostrar**

Introducción a Java EE

- Cliente-servidor y protocolo HTTP
- El servidor Tomcat
- Introducción a las aplicaciones web Java EE
- Servlets y JSP
- Visión global de Java EE

Arquitectura multi-capa



APIs y tecnologías