CC5002 - Desarrollo de Aplicaciones Web

Clase 10: Java Standard Tag Libraries

Plan de clases proyectado (1era Mitad)

1. Intro
2. HTTP y HTML
3. Servlets y Tomcat
4. JSP y Threads
5. Sesiones y Cookies
6. Bases de datos
7. JavaBeans
8. JSP
9. MVC
10. ->JSTL
11. Seguridad
12. CSS
13. Javascript
14. AJAX
15. Control

Clase de Hoy

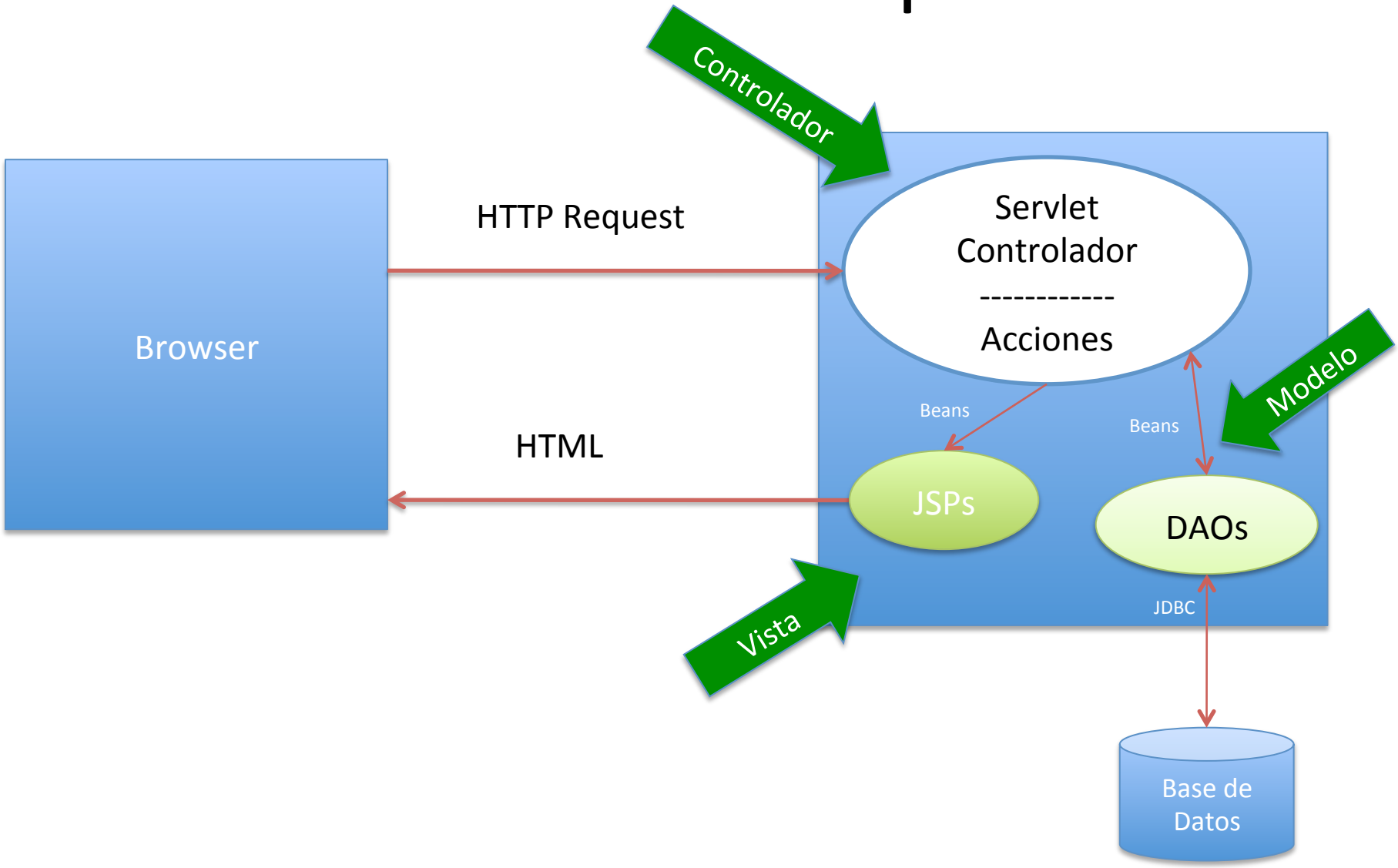
- Temas Administrativos
- Preguntas
- JSTL

Preguntas?

Pregunta Para Uds.

- Como van con la tarea 4?
- Muy fácil?
- Deberíamos incrementarle la dificultad?

Tarea 4: MVC es Requerido



Pasos para Crear una Aplicación Web MVC

- Modelo
  - Esquema de la base de datos
  - Definir beans y DAOs (Implementado con BeanFactory?)
- Controlador
  - Son muy parecidos todos
  - Personalizar: inicialización, chequeos de seguridad, ubicación de las vistas.
- Acciones
  - Procesan/Validan los parámetros del request (Usando FormBeanFactory?)
  - Llaman a un DAO para leer/escribir en la base de datos
  - Si hay error, setear atributos, seleccionar vista
  - Si no hay error, setear atributos, seleccionar vista
- Vistas
  - JSPs, JSTL, EL
  - Templates (<jsp:include>)

Cuando una webapp es MVC?

✓Despacha primero a un servlet?

✓Servlet despacha a un JSP para el formateo?

✓Servlet utiliza DAOs para el acceso a los datos.

¿Servlet no hace formateo?

☺Servlet no genera tags HTML

¿JSP no procesa lógica de la aplicación?

☺JSP no contiene construcciones Java

Meta

✓Usar Modelo-Vista-Controlador

✓Servlet no genera tags HTML

¿JSP no contiene expresiones java?

- Podemos usar tag libraries
- Tags especiales en el JSP que se mapean en código Java el cual es ejecutado para generar la salida HTML

Tag Libraries

- Jakarta Tag Libraries
  - 29 tag libraries + JSTL
- JSTL – Java Standard Tag Library
  - Core: if, forEach, etc.
  - Internacionalización, XML, SQL
- Escribe tus propias
  - jsptags.com
    - Noticias sobre tag libraries
    - Tag libraries para descargar

JSTL: Java Standard Tag Library

- Descargable desde:
  - http://jakarta.apache.org

- La versión actual es jakarta-taglibs-standard-1.1.2.zip
  - Descargar de un mirror:
  - http://jakarta.apache.org/site/downloads/downloads_taglibs-standard.cgi

  - Para los core tags (Esenciales): Colocar jstl.jar y standard.jar en lib
  - Declarar el uso en el comienzo de los jsp
  - Tener el tag <webapp> correcto en el web.xml

- Documentación
  - http://java.sun.com/products/jsp/jstl/1.1/docs/tlddocs/index.html

JSTL Core Tags

- Tags Condicionales: `if, choose/when`
- Tags de loop: `forEach, forTokens`
  - El tag forEach itera sobre un arreglo java o sobre un java.util.Collection

JSTL Expression Language (EL)

- Permite evaluar expresiones simple
- Las variables EL permiten referenciar en contextos de pagina, sesiones, requests, y aplicación.
  - Ej: `${list}` se convierte como `request.getAttribute("list")`
  - Ej: `${list.date}` devuelve `list.getDate()`
- EL tiene un método `empty ()` que verifica si un string es null o vacío.

“EL” es parte de JSP 2.0

- EL fue tan popular, que lo movieron dentro de los specs de JSP.
  - Si declaran su aplicación como JSP 2.0 (Usando el tag <webapp> correcto en el web.xml) pueden usar EL en sus JSPs.

Ejemplo JSTL/EL

- Usando el tag JSTL <c:forEach>
- Usando EL

```
<c:forEach var="person" items="${personList}">
   <tr>
    <td>${person.lastName}</td>
    <td>${person.firstName}</td>
   </tr>
</c:forEach>
```

Tags de Formateo JSTL

- Para formatear fechas y números
- Ejemplo

```
<fmt:formatDate
   value="${entry.updated}"
   type="both"
   pattern="MMM-dd-yyyy h:mmaa"
/>
```

Ejemplo

- Address Book
  - Mantiene una libreta de direcciones compartida en un sitio web.
- Cosas a notar:
  - MVC con JSTL en JSPs
  - User Bean: Password son hasheados. (Al igual que en Photolicious)
  - Entry Bean: Muchas propiedades
  - Uso de GenericDAO (Nuevo a partir de BeanFactory 2.3.2)
  - Controller y Actions tienen requerimientos inusuales de seguridad
  - Control de concurrencia usando digests

Comparen con los JSP del Ejemplo Photolicious

- list.jsp vs list-entries.jsp

Noten el Formateo de las Fechas

- Ejemplos:
  - display-entry.jsp
  - entry-entry.jsp
  - entry-form.jsp

El Estilo se Mantiene

- Servlet Controlador
  - Inicialización, seguridad, despacho a acción y vistas
- Action
  - Chequea errores en el input
  - Acceso a la base de datos por medio de DAOs
  - Setea atributos en el request (O en la sesión en algunos casos)
  - Selecciona la vista (O ocasionalmente redirecciona)
- DAO
  - Beans, BeanFactory
- Vistas
  - Genera el HTML
  - Obtiene el contenido dinámico de los atributos.

¿Qué es lo difícil en la Tarea 4?

- Esquema de la BD/Código de acceso
  - Beans/Factories
- Servlet Controlador
  - copy & paste
- Actions
  - También pueden copiar unas
- Vistas
  - JSP, JSTL, EL

Desventajas de HTML en Java (Servlets)

- Necesidad de escapar el HTML
- No es comodo hacer out.println por cada linea de código.
- No se puede usar una herramienta WYSIWYG para editar el HTML (Dreamweaver)

Desventajas de Java en JSP (HTML)

- Enredado hacer seguridad, validación y lógica de negocios dentro de un JSP
  - Mucho código repetido
  - Difícil de mantener

¿Por qué estamos viendo esta esto?

- Somos todos adultos
- Todos somos capaces de escribir código Java
- Realmente necesitamos agregar un mecanismo adicional para separar Java del HTML?

Ventajas de la Separación

- Facilidad de mantención
  - Lógica en un solo lugar
  - Presentación en otro lugar.

Ventajas de la Separación (2)

- Mayor facilidad para cambiar la presentación sin afectar la lógica de la aplicación
  - Los sitios suelen cambiar a menudo su presentación

Ventajas de la Separación (3)

- Diferentes personas trabajan en un proyecto grande
  - La gente de la lógica de la aplicación, son gente mas técnica
  - La gente de la presentación, son mas artistas

Próxima Clase

- Seguridad