DESARROLLO DE APLICACIONES WEB

Tema 5.- Java Entreprise Edition

Cookies, Sesiones y Almacenamiento Local

Contenido

- Introducción
- Cookies
- Sesiones
- Almacenamiento Local (JavaScript)
  - localStorage
  - sessionStorage

Introducción

- Protocolo HTTP es un protocolo sin estado
- Las peticiones son independientes unas de otras
- Esto, que supone una ventaja: por su simplicidad y bajo consumo de ancho de banda, acaba convirtiéndose en un inconveniente, por la imposibilidad de conocer qué cliente realiza la petición y por tanto su estado entre peticiones.
- Soluciones:
  - Reescritura URL o Campos Ocultos
  - Cookies
  - Sesiones
  - Almacenamiento Local

Contenido

- Introducción
- Cookies
- Sesiones
- Almacenamiento Local (JavaScript)
  - localStorage
  - sessionStorage

Cookies

- Son pares name=value que se **almacenan en el cliente** (mediante una cabecera HTTP de respuesta – set-Cookie) y son **enviadas en cada petición al servidor** (mediante una cabecera de petición – Cookie).
- Las cookies tienen las siguientes propiedades:
  - Name: Nombre con el que se almacena y accede
  - Value: Valor asignado
  - Expired: valor (seg) hasta el que estará activa la cookie y seguirá enviándose al servidor
  - Domain: Servidor al que se enviará la cookie
  - Path: Ruta dentro del dominio al que se enviará la cookie
  - Security: Valor booleano que indica si la cookie requiere un protocolo seguro (SSL) para su envío.
  - Comment: Comentario que describe la utilidad o propósito de la cookie

Cookies en Java

Ver javadoc API

Constructor
`Cookie(String name, String value)`

Método	
`String`	**`getName()`**, **`getValue()`**, **`getComment()`**, **`getDomain()`**, **`getPath()`** Devuelve el nombre, el valor, la descripción, dominio y la ruta de la cookie.
`void`	**`setValue(String newValue`**, **`setComment(String purpose)`**, **`setDomain(String domain)`**, **`setPath(String uri)`** Asigna el valor, el comentario (propósito), dominio y la ruta
`int`	**`getMaxAge()`** Devuelve duración, en segundos, de la cookie
`void`	**`setMaxAge(int expiry)`** Asigna el tiempo, en segundo, durante el cual la cookie estará disponible
`boolean`	**`getSecure()`** Devuelve true si la cookie sólo se enviará por HTTPS o SSL
`void`	**`setSecure(boolean flag)`** Establece (parámetro true) si la cookie sólo se enviará por HTTPS or SSL.

Cookies en Java

Ver javadoc API

Uses of Class
jakarta.servlet.http.Cookie

Packages that use Cookie

Package	Description
jakarta.servlet.http	The jakarta.servlet.http package contains a number of classes and interfaces that describe and define the contracts between a servlet class running under the HTTP protocol and the runtime environment provided for an instance of such a class by a conforming servlet container.

Uses of Cookie in jakarta.servlet.http

Methods in jakarta.servlet.http that return Cookie

Modifier and Type	Method	Description
`Cookie[]`	**HttpServletRequest.**`getCookies()`	Returns an array containing all of the `Cookie` objects the client sent with this request.
`Cookie[]`	**HttpServletRequestWrapper.**`getCookies()`	The default behavior of this method is to return getCookies() on the wrapped request object.

Methods in jakarta.servlet.http with parameters of type Cookie

Modifier and Type	Method	Description
`void`	**HttpServletResponse.**`addCookie(Cookie cookie)`	Adds the specified cookie to the response.
`void`	**HttpServletResponseWrapper.**`addCookie (Cookie cookie)`	The default behavior of this method is to call addCookie(Cookie cookie) on the wrapped response object.

Cookies en Java

- Para Crear y Enviar Cookies en la respuesta HTTP:
  - 1) Cookie c = new Cookie(“nombre”, “valor”);
  - 2) Asignar fecha de expiración: segundos que deben pasar hasta que caduque, Domain, Path, Comment, etc

    c.setMaxAge(seg_expiry) // día = 60 x 60 x 24
  - 3) Añadir la cookie como cabecera de la respuesta:

    response.addCookie(c);

Cookies en Java

- Para leer las Cookies en la petición HTTP:
  - 1) Leer el array de cookies desde la petición: request.getCookies();

    Cookie[] cookies;

    cookies = request.getCookies();
  - 2) Procesar cada cookie.

    for (Cookie c: cookies) { … } // getName(), getValue(), getComment(), getDomain(), getPath()

    En las vistas, si fuese necesario podemos acceder mediante EL

    ${cookie[“name”].value}

Cookies en Java: ejemplo

```
Cookie[] cookies = request.getCookies();
 Cookie visitas = null;
 if (cookies != null) {
     for (Cookie c : cookies) {
         if (”visitas".equals(c.getName())) {visitas = c; }
     }
 }
 int v = 0;
 if (visitas != null) {
     // La cookie existe. Aquí se trata
     v = Integer.parseInt(visitas.getValue());
 }
 v = v + 1;
 Cookie c = new Cookie(”visitas”, String.valueOf(v));
  int expiry = 86400; // (1día) = 24 x 60 x 60 segundos para que caduque
  c.setMaxAge(expiry);
  response.addCookie(c);
```

Cookies en JavaScript

- Las Cookies puede crearse y manejarse desde JavaScript

```
function setCookie(c_name,value,exdays) {
        var exdate=new Date();
        exdate.setDate(exdate.getDate() + exdays);
        var c_value=escape(value) + ((exdays==null) ? "" : ";
                                        expires="+exdate.toUTCString());
        document.cookie=c_name + "=" + c_value;
}
```

Cookies en JavaScript

- Las Cookies puede obtenerse desde JavaScript

```
<script type="text/javascript">
        var allCookies = document.cookie;
        var arrayCookies  = allCookies.split(';');
        for(var i=0; i<arrayCookies.length; i++){
            var name = arrayCookies[i].split('=')[0];
            var value = arrayCookies[i].split('=')[1];
            document.write("Cookie: " + name + " = " + value);
        }
 </script>
```

Cookies: Codificación

- Las Cookies pueden ir codificadas (esto no quiere decir encriptadas)

```
import java.util.Base64;
…
String config = “datos a guardar";
String encodedConfig = Base64.getEncoder().encodeToString(config.getBytes());
Cookie c = new Cookie("config", encoded);
response.addCookie(c);
…
byte[] configBytes = Base64.getDecoder().decode(c.getValue());
String config = new String(configBytes);
```

- En JavaScript
  - btoa(cad): Devuelve la cadena codificada
  - atob(cadbase64): devuelve la cadena decodificada

Contenido

- Introducción
- Cookies
- Sesiones
- Almacenamiento Local (JavaScript)
  - localStorage
  - sessionStorage

Sesiones

- Las sesiones permiten mantener información, almacenando objetos, entre diferentes peticiones de un mismo usuario.
  - setAttribute(name, obj)
  - getAttribute(name)
  - removeAttribute(name)

- La sesión persiste durante un tiempo determinado definido en el Servidor o en la aplicación, o bien, cuando el navegador se cierra.
- La sesión se mantiene como una cookie en el cliente (guardando su identificador) y los objetos son ubicados en el servidor.

Sesiones en Java

- Interface HttpSession
  - Servlets: HttpSession session = request.getSession();
  - JSP: Objeto session
  - EL: ${sessionScope.att}

Sesiones en Java

Ver javadoc API

Métodos	
void	**setAttribute**(**String** name, **Object** value) Registra un objeto en la sesión con el nombre especificada
Object	**getAttribute**(**String** name) Devuelve el objeto registrado en la sesión o null si no existe (casting)
Enumeration<String>	**getAttributeNames**() Devuelve un Enumeration de String con los nombre de todos los objetos de la sesión.
void	**removeAttribute**(**String** name) Elimina de la sesión el objeto con el nombre especificado.
String	**getId**() Devuelve el identificador asignado a la sesión.
boolean	**isNew**() Devuelve true si el cliente aún no conoce la sesión false en caso contrario.
void	**invalidate**() Invalida la sesión y borra todos los objetos de la misma
void	**setMaxInactiveInterval**(int interval) Establece el tiempo, en segundos, entre peticiones de duración de la sesión.
int	**getMaxInactiveInterval**() Devuelve el tiempo, en segundos, que el Servidor mantendrá la sesión.
long	**getCreationTime**() Devuelve el tiempo cuando se creó la sesión (milisegundos desde 1/1/1970).
long	**getLastAccessedTime**() Devuelve el tiempo transcurrido desde la última petición. (mlgs 1/1/1970)
ServletContext	**getServletContext**() Devuelve el contexto, ServletContext, donde la sesión está activa.

Sesiones Java: Ej simple Carrito Compra

Ejemplo: Vista artículos.jsp - Listado de Artículos

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Productos</h1>

        <p><a href="/agenda/productos?cod=1">Producto 1</a></p>
        <p><a href="/agenda/productos?cod=2">Producto 2</a></p>
        <p><a href="/agenda/productos?cod=3">Producto 3</a></p>
        <p><a href="/agenda/productos?cod=4">Producto 4</a></p>


    </body>
</html>
```

Sesiones Java: Ej simple Carrito Compra

Ejemplo: Controlador para guardar artículos y ver el carrito

```
HttpSession session = request.getSession();
String vista = "error";
String accion = request.getServletPath();

if (accion.equals("/productos")) {
    List<String> productos;
    String codigo = request.getParameter("cod");
    // POR SEGURIDAD: Comprobar que llega y es correcto
    if (codigo != null) {
          if(session.getAttribute("productos")==null) productos = new ArrayList();
          else productos = (List<String>) session.getAttribute("productos");
          productos.add(codigo);
          session.setAttribute("productos", productos);
    }
    vista = "productos";
} else if (accion.equals("/carrito")) {
    vista = "carrito";
}

request.setAttribute("view", vista);
RequestDispatcher rd = request.getRequestDispatcher("/WEB-INF/views/template.jsp");
rd.forward(request, response);
```

Sesiones Java: Ej simple Carrito Compra

Ejemplo: vista carrito.jsp para mostrar artículos en el carrito

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<c:if test="${!empty sessionScope.productos}">

    <c:forEach var="art" items="${sessionScope.productos}">

        <p>Artículo ${art}</p>

    </c:forEach>


</c:if>

<c:if test="${empty sessionScope.productos}">

No hay artículos en el carrito

</c:if>
```

Contenido

- Introducción
- Cookies
- Sesiones
- Almacenamiento Local (JavaScript)
  - localStorage
  - sessionStorage

Almacenamiento Local

- HTML5/JavaScript
- Objetos “localStorage” y “sessionStorage” del objeto window.
- sessionStorage actúa sobre el ámbito de la sesión del cliente, cuando el usuario cierra el navegador los datos se pierden,
- localStorage los datos perduran en el navegador hasta que sean eliminados.

Almacenamiento Local

- Ventajas frente a Cookie/sesiones
  - Almacenamiento en el Cliente (reducción de espacio en el servidor – mayor escalabilidad)
  - Envío entre cliente y servidor más eficiente, solo cuando sea necesario (reduce ancho de banda y reduce problemas de seguridad)

Almacenamiento Local

- Métodos de “localStorage” y “sessionStorage” para almacenamiento y recuperación de datos:
  - getItem(key): devuelve un string con el valor del elemento con clave key.
  - setItem(key, value): almacena un valor (value) referenciado por una clave (key).
  - removeItem(key): elimina el par clave/valor con clave igual a key.
  - length: atributo que contiene el número de elementos (pares clave/valor) almacenados.
  - key(index): devuelve un string con la clave (no el valor) del elemento que ocupe la posición index dentro de la colección de datos.
  - clear(): elimina todos los elementos.

  - Se puede acceder como hash: localStorage[key] o sessionStorage[key]

Almacenamiento Local

- Ejemplos: (idem para sessionStorage)

Soporte por parte del navegador

```
if (window["localStorage"]) {
    //SI soporta almacenamiento local
}else{
   //NO soporta almacenamiento local
  alert('Tu navegador es demasiado viejo, ¡actualízate!');
}
```

Guardar un elemento

```
localStorage.setItem(”saludo", ”Hola!");
```

Obtener un elemento

```
var tarea = localStorage.getItem(”saludo");
```

Almacenamiento Local

- Ejemplos: (idem para sessionStorage)

Datos no String

```
localStorage.setItem('numero',5);

let miNumero = parseInt(localStorage.getItem('numero');

localStorage.setItem('precio',9.99);

let precio = parseFloat(localStorage.getItem('precio'));
```

Almacenamiento Local

- Ejemplos: (idem para sessionStorage)
  - Acceso mediante [ ]

    localStorage[‘color'] = red’;

    let color = localStorage[‘color'];
  - Longitud y Key

```
for (var i = 0; i < localStorage.length; i++) {
    let clave = localStorage.key(i);
    let valor = localStorage[clave];
    alert(valor);
}
```

  - Eliminar

    localStorage.removeItem(“key”);

    localStorage.clear();