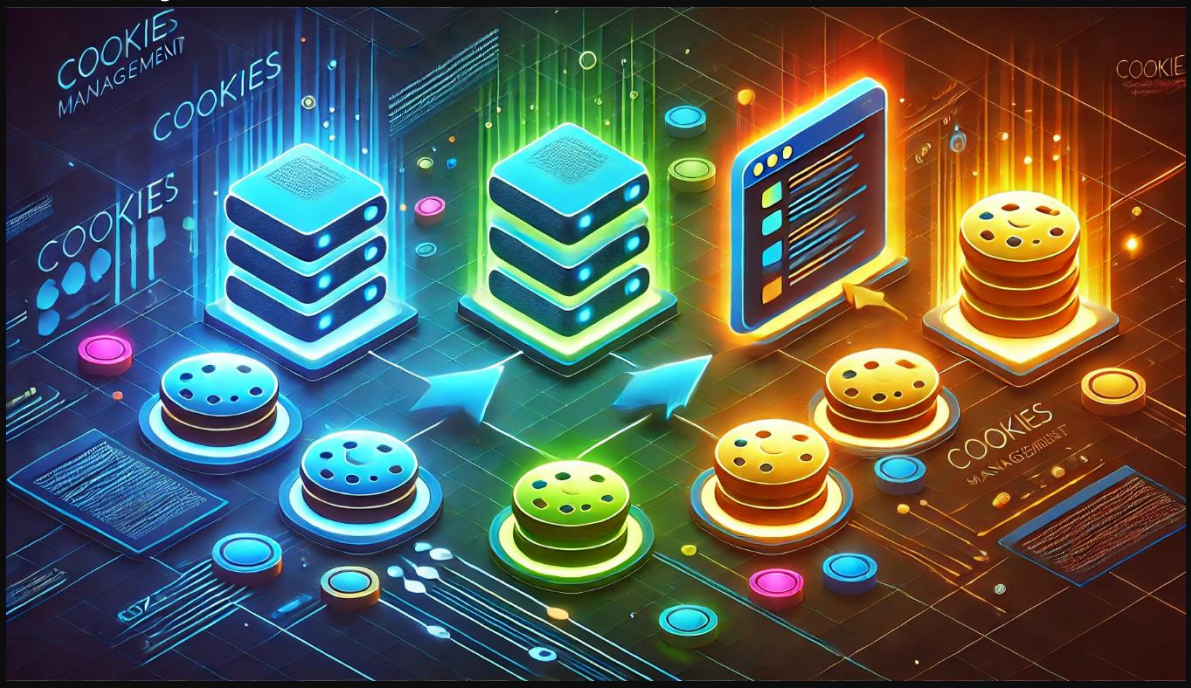
Manejo de Cookies con Servlets



Guía paso a paso: Proyecto "Manejo de Cookies " en Java, NetBeans y GlassFish

1. Descripción de la aplicación

La aplicación consta de un archivo HTML sencillo que contiene un enlace a un servlet. El servlet utiliza cookies para detectar si un usuario es nuevo o recurrente en el sitio. Si es nuevo, se creará una cookie; si no, se mostrará un mensaje para indicar que el usuario ya ha visitado anteriormente la página.

2. Creación del Proyecto en NetBeans

1. **Abrir NetBeans.**
2. **Crear un nuevo proyecto:**
   - Selecciona `File > New Project`.
   - Elige `Java with Maven > Web Application`.
   - Nombre del proyecto: `ManejoCookies`.
   - Selecciona GlassFish como servidor.
   - Usa Jakarta EE para el perfil del proyecto.

3. Archivos necesarios

a) Código HTML (index.html)

Este archivo servirá como la página principal con un enlace al servlet que gestiona las cookies.

1. **Crear el archivo HTML**:
    - Ve a la carpeta `webapp`.
    - Crea un archivo llamado `index.html` con el siguiente contenido:

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Ejemplo de Cookies</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <h1>Ejemplo de Cookies</h1>
        <a href="/ManejoCookies/CookiesServlet">Link al servlet de manejo de cookies</a>
    </body>
</html>
```

b) Servlet de Cookies (CookiesServlet.java)

Este servlet maneja las cookies. Detecta si el usuario es nuevo o si ya ha visitado la página anteriormente, y actualiza el contador de visitas.

1. **Crear el archivo Servlet**:
    - En la carpeta `java`, dentro de tu paquete principal, crea un archivo llamado `CookiesServlet.java` con el siguiente contenido:

```java
package web;

import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.*;

@WebServlet("/CookiesServlet")
public class CookiesServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
IOException {
        // Suponemos que el usuario visita por primera vez nuestro sitio
```

```
        boolean nuevoUsuario = true;

        // Obtenemos el arreglo de cookies
        Cookie[] cookies = request.getCookies();

        // Buscamos si ya existe una cookie creada con anterioridad
        // llamada visitanteRecurrente
        if (cookies != null) {
            for (Cookie c : cookies) {
                if (c.getName().equals("visitanteRecurrente") && c.getValue().equals("si")) {
                    // Si ya existe la cookie, es un usuario recurrente
                    nuevoUsuario = false;
                    break;
                }
            }
        }

        String mensaje;
        if (nuevoUsuario) {
            // Creamos una nueva cookie para identificar al usuario recurrente
            Cookie visitanteCookie = new Cookie("visitanteRecurrente", "si");

            // Mejora: puedes establecer la cookie como segura y solo accesible vía HTTP
            // Si el sitio utiliza HTTPS, la siguiente línea asegura que la cookie solo se envíe a
través de HTTPS
            // visitanteCookie.setSecure(true);
            // La cookie no será accesible desde JavaScript, mejorando la seguridad
            visitanteCookie.setHttpOnly(true);

            response.addCookie(visitanteCookie);
            mensaje = "Gracias por visitar nuestro sitio por primera vez";
        } else {
            mensaje = "Gracias por visitar NUEVAMENTE nuestro sitio";
        }

        // Especificamos el tipo de contenido como HTML y establecemos el charset
        response.setContentType("text/html;charset=UTF-8");

        // Usamos try-with-resources para asegurarnos de que el PrintWriter se cierra correctamente
        try (PrintWriter out = response.getWriter()) {
            out.print("Mensaje: " + mensaje);
        }
    }
}
```

Explicación:

- Este servlet revisa si hay cookies en la solicitud del cliente.
- Si encuentra una cookie llamada `visitanteRecurrente`, actualiza la variable usuarioNuevo a false.
- Si no encuentra la cookie, asume que el usuario es nuevo y crea una nueva cookie.
- **`setHttpOnly(true)`**: protege la cookie de ser accesible por JavaScript, lo que mejora la seguridad al prevenir ataques de tipo Cross-Site Scripting (XSS).
- **Opción `setSecure(true)`** (comentada): es recomendable si tu aplicación utiliza HTTPS, ya que solo enviaría la cookie a través de una conexión segura. Puedes activarlo si estás usando HTTPS.

4. Conclusión

En esta guía hemos creado una aplicación sencilla en Java con NetBeans, GlassFish y Jakarta EE que demuestra cómo manejar cookies utilizando servlets. La aplicación distingue entre usuarios nuevos y recurrentes a través del uso de cookies, lo que permite personalizar la experiencia del usuario basándose en las visitas anteriores.

Saludos!

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx