Tarea: Investigación

Christofer Fabián Chávez Carazas
Universidad Nacional de San Agustín de Arequipa
Escuela Profesional de Ciencia de la Computación
Computación Centrada en Redes

22 de diciembre de 2017

Enunciado

Realizar una investigación sobre:

- API de Servlets de Java
- Servlets HTTP
- Set-Cookie
- Coookie
- Carrito de compra

1. **API de Servlets de Java**

   Es una tecnología que nos permite crear aplicaciones web interactivas (dinámicas), es decir, le permite al usuario interactuar con la aplicación (hacer consultas, insertar y eliminar datos, etc). Un Servlet es un objeto java que pertenece a una clase que extiende de javax.servlet.http.HttpServlet. Son pequeños programas escritos en Java que admiten peticiones a través del protocolo HTTP. Los servlets reciben peticiones desde un navegador web, las procesan y devuelven una respuesta al navegador, normalmente en HTML. Para realizar estas tareas podrán utilizar las clases incluidas en el lenguaje Java. Estos programas son los intermediarios entre el cliente (casi siempre navegador web) y los datos (BBDD).
   Los servelts pueden ser incluidos en servidores que soporten la API de Servlet (ver servidores). La API no realiza suposiciones sobre el entorno que se utiliza, como tipo de servidor o plataforma, ni del protocolo a utilizar, aunque existe una API especial para HTTP. Los Servlets son un reemplazo efectivo para los CGI en los servidores que los soporten ya que proporcionan una forma de generar documentos dinámicos utilizando las ventajas de la programación en Java como conexión a alguna base de datos, manejo de peticiones concurrentes, programación distribuida, etc. Se suelen utilizar cuando:

   - Las páginas web se basa en datos que proporciona el usuario. Por ejemplo, e-commerce sites.

- Los datos cambian frecuentemente. Por ejemplo, Webs meteorológicas.
- La página web utiliza información de bases de datos u otras fuentes. Por ejemplo, Aplicaciones comerciales.

Tienen las siguientes propiedades:

- **Manejo de Sesiones:** Se puede hacer seguimiento de usuarios a través de distintos servlets a través de la creación de sesiones.
- **Utilización de Cookies:** Las cookies son pequeños datos en texto plano que pueden ser guardados en el cliente. La API de servlets permite un manejo fácil y limpio de ellas.
- **Multi-thread:** Los servlets soportan el acceso concurrente de los clientes, aunque hay que tener especial cuidado con las variables compartidas.
- **Programación en Java:** Se obtienen las características de multiplataforma o acceso a APIs como JDBC, RMI, etc.

Los servlets utilizan clases e interfaces de dos paquetes: javax.servlet y javax.servlet.http. El paquete javax.servlet contiene clases para admitir servlets genéricos independientes del protocolo. Estas clases se extienden por las clases en el paquete javax.servlet.http para agregar funcionalidad específica de HTTP. El nombre del paquete de nivel superior es javax en lugar del familiar java, para indicar que la API del servlet es una extensión estándar.
Cada servlet debe implementar la interfaz javax.servlet.Servlet. La mayoría de los servlets se implementan extendiendo una de dos clases especiales: javax.servlet.GenericServlet o javax.servlet.http.HttpServlet. Un servlet independiente del protocolo es una subclase de GenericServlet, mientras que uns servlet HTTP es una subclase HttpServlet, que a su vez es una subclase de GenericServlet con funcionalidad adicional específica de HTTP.
A diferencia de un programa Java normal, y al igual que un applet, un servlet no tiene un método *main()*. En cambio, ciertos métodos de un servlet son invocados por el servidor en el proceso de manejo de solicitudes. Cada vez que el servidor envía una solicitud a un servlet, invoca el método *service()* del servlet.
Un servlet genérico debe sobre escribir su método *service()* para manejar las solicitudes según corresponda para el servlet.

```
public abstract void service(ServletRequest request, ServletResponse response)
        throws ServletException, java.io.IOException
```

El método *service()* acepta dos parámetros: un objeto de solicitud y un objeto de respuesta. El objeto de solicitud le dice al servlet acerca de la solicitud, mientras que el objeto de respuesta se usa para devolver una respuesta. En la siguiente figura se muestra a una servlet genérico manejando una petición.

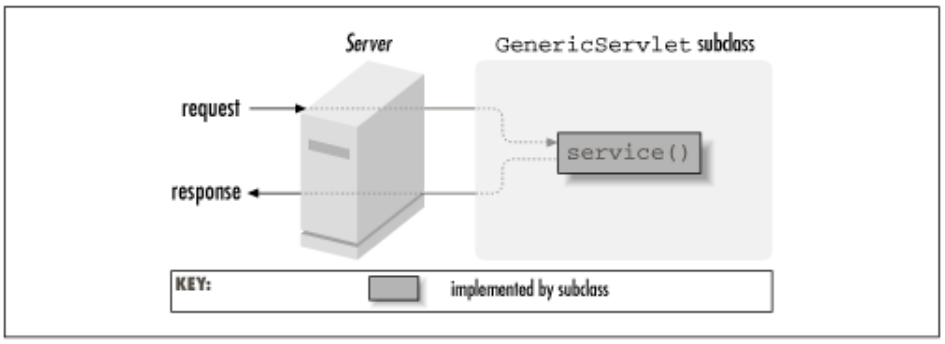


Figura 1: Un servlet genérico manejando una petición

2. **Servlets HTTP**

Se se quiere crear un Servlet HTTP se debe extender la clase javax.servlet.http.HttpServlet. A diferencia del servlet genérico, el servlet HTTP no sobre escribe el método *service()* En cambio debe sobre escribir al menos uno de los siguientes métodos:

- **doGet():** El método service del servlet llama a este método para gestionar la solicitud HTTP GET del cliente. El método Get se utiliza para obtener información del servidor.
- **doPost():** Se utiliza para enviar información al servidor.
- **doPut():** Este método es similar al *doPost* pero con la diferencia de que se envía archivos al servidor en vez de información.
- **doDelete():** Permite que un cliente elimine un documento, página web o información del servidor.
- **init() y destroy():** Se utilizan para administrar recursos que se mantienen durante la vida del servlet.
- **getServletInfo():** Devuelve información sobre el servlet como el autor, la versión y derechos de autor.

En la siguiente figura se muestra un servlet HTTP manejando peticiones GET y POST:

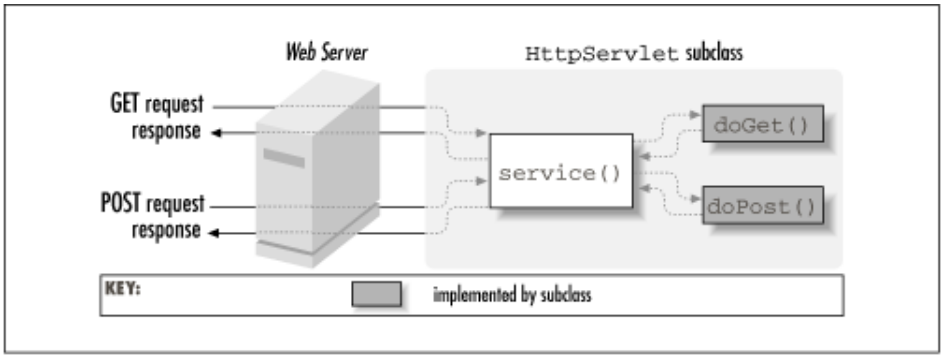


Figura 2: Servlet Http manejando peticiones GET y POST

Un pequeño servlet de ejemplo es el siguiente:

```
public class SimpleServlet extends HttpServlet {

    // Maneja el émtodo GET de HTPP para
    // construir una sencilla pagina Web.

    public void doGet (HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
    PrintWriter out;
    String title = "Simple Servlet Output";

    // primero selecciona el tipo de contenidos y otros campos de cabecera de la respuesta
    response.setContentType("text/html");
    // Luego escribe los datos de la respuesta
    out = response.getWriter();
    out.println("<HTML><HEAD><TITLE>");
    out.println(title);
    out.println("</TITLE></HEAD><BODY>");
    out.println("<H1>" + title + "</H1>");
    out.println("<P>This is output from SimpleServlet.");
    out.println("</BODY></HTML>");
    out.close();
    }
}
```

Este método recibe los parámetros dados por el cliente a través de la clase *HttpServletRequest* y encapsula la respuesta que se le dará al cliente a través de la clase *HttpServletResponse*. El servlet puede retornar al cliente cualquier tipo de información, desde texto plano hasta un ejecutable, por lo que es necesario señalar inicialmente qué tipo de respuesta se dará a través del método *setContentType*. Luego se obtiene el objeto para poder escribir texto al cliente a través del método *getWriter* con el cual se puede retornar una página web llamado sucesivamente el método *println* hasta terminar con *close*.

3. **Set-Cookie**

   La cabecera de respuesta HTTP Set-Cookie se usa para enviar cookies desde el servidor al agente de usuario. Tiene las siguiente sintaxis:

$$Set - Cookie : < cookie - name >=< cookie - value >; < directives >$$

   Las directivas disponibles son las siguientes:

   - <cookie-name>=<cookie-value>

     Una cookie comienza con un par nombre-valor:

     - Un <cookie-name> puede ser cualquier cosa excepto caracteres de control (CTLs) o espacios y tabulaciones. Tampoco debe contener caracteres de separación.
     - Para un <cookie-value> Se permite usar cualquier caracter US-ASCII excluyendo CTLs, espacios en blanco, comillas dobles, comas, punto y coma y la barra invertida.
     - **Prefijo __Secure-:** Para cookies provenientes de páginas HTTPS.
     - **Prefijo __Host-:** Para cookies provenientes de páginas HTTPS y no son enviadas a subdominios.

- Expires=<date>

  El tiempo de vida útil máxima de la cookie. Si no se especifica, la cookie tendrá la vida útil de una cookie de sesión, es decir, la cookie se eliminará cuando se cierra la sesión.

- Max-Age=<non-zero-digit>

  Número segundos hasta que la cookie expira.

- Domain=<domain-value>

  Se especifica los hosts a los que se envía la cookie. Si no se especifica, se establece de manera predeterminada en la parte del host de la ubicación del documento actual.

- Path=<path-value>

  Indica una ruta de URL que debe existir en el recurso solicitado antes de enviar el encabezado de Cookie.

- HttpOnly

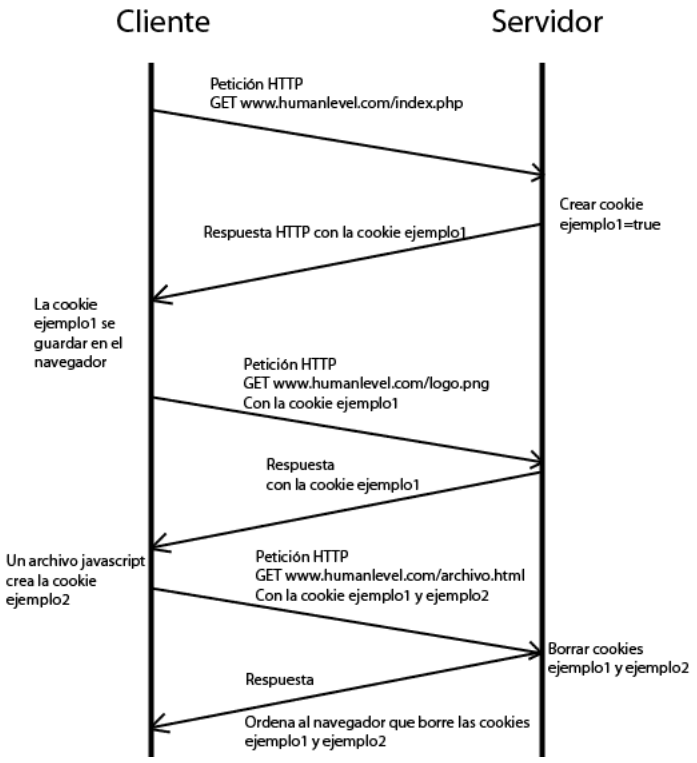
  Las cookies HTTP-only no son accesibles a través de JavaScript a través de la propiedad Document.cookie, las API XMLHttpRequest y Request para mitigar ataques contra scripts de sitios cruzados (XSS).

4. **Cookie**

   Una cookie es un fichero que se descarga en un ordenador al acceder a determinadas páginas web. Las cookies permiten a una página web, entre otras cosas, almacenar y recuperar información sobre los hábitos de navegación de un usuario o de su equipo y, dependiendo de la información que contengan y de la forma en que utilice el equipo, pueden utilizarse para reconocer la sesión del usuario. En general pueden hacer lo siguiente:

   - Recordar el usuario y contraseña en una página web.
   - Mostrar publicidad online en función de tus intereses.
   - Obtener estadísticas para un webmaster.
   - Recordar preferencias en una web.
   - Compartir en redes sociales.
   - Llenar un carrito de compra en una tienda online.

   En la siguiente imagen se muestra un ejemplo de lo que ocurre cuando se crean cookies en el cliente y el servidor.



Las cookies se pueden dividir en diferentes tipos:

- **Según quien las gestione:**
  - **Cookies Propias:** Son aquellas que se envían al equipo terminal del usuario desde un equipo o dominio gestionado por el propio editor y desde que se presta el servicio solicitado por el usuario.
  - **Cookies de Terceros:** Son aquellas que se envían al equipo terminal del usuario desde un equipo o dominio que no es gestionado por el editor, sino por otra entidad que trata los datos obtenidos a través de las cookies.
- **Según su finalidad:**
  - **Cookies Técnicas:** Son aquellas que permiten al usuario la navegación a través de la página web, plataforma o aplicación y la utilización de las diferentes opciones o servicios que en ella existen.
  - **Cookies de Análisis:** Son aquellas que permiten al responsable de las mismas, el seguimiento y análisis del comportamiento de los usuarios de los sitios web a los que están vinculadas.
- **Según su tiempo de vida:**
  - **Cookies de Sesión:** Son las diseñadas para recabar y almacenar datos mientras el usuario accede a una página web. Se suelen emplear para la prestación del servicio solicitado por el usuario en una sola ocasión y se elimina cuando el usuario cierra la sesión.

- **Cookies Persistentes:** Son las diseñadas para que los datos sigan almacenados en el terminal y pueden ser accedidos y tratados durante un periodo definido por el responsable de la cookie, y que puede ir de unos minutos a varios años

5. **Carrito de compra**

   Desde que Internet se convirtió en un escenario vital para las ventas y el marketing, la tecnología web se ha orientado en los últimos tiempos a potenciar y adecuar sus herramientas para hacer de los negocios online una alternativa fiable y productiva. En ese sentido, ningún recurso tan efectivo como el carrito de compras para viabilizar el comercio virtual y poner a empresas y clientes en un nuevo nivel de acercamiento.

   Las 4 fortalezas o características principales de la mercadotecnia en Internet:

   - **Flujo:** Denominado así porque genera una relación constante entre ambas partes. Según Fleming, flujo es "el estado mental en que entra un usuario de Internet al sumergirse en una web que le ofrece una experiencia llena de interactividad y valor añadido". Tal suma de sensaciones propicia una mayor facilidad de acercamiento entre las partes, potenciando las oportunidades de alcance de contenidos y ofrecer productos. Todo entra por la vista.
   - **Funcionalidad:** Porque la oportunidad de acercarse hace posible un futuro proceso de venta. Si el cliente ha entrado en estado de flujo, quiere decir que está en camino de ser captado, pero para que el flujo de la relación no se rompa, queda dotar a la presencia on-line de funcionalidad, es decir, construir páginas atractivas, con navegación clara y útiles para el usuario. En posteos anteriores, hemos hablado de aquello, es decir, de la necesidad de construir websites visibles al usuario maximizando su experiencia de usuario para fidelizarlo. Para una tienda online resulta imprescindible este atributo.
   - **Feedback:** Porque permite el interés e intercambio de información. El usuario está en estado de flujo y no se exaspera en su navegación, por lo que las opciones de elegir el carrito de compras crecen. El website da entonces la oportunidad de preguntar al cliente qué le gusta y qué le gustaría mejorar, y nutrirse de esa información para mejorar. Dialogar con el cliente para conocerlo mejor y construir una relación basada en sus necesidades refuerza el feedback, probablemente, una de las mayores fortalezas de este negocio.
   - **Fidelización:** Porque permite un vínculo duradero y especial. Internet y sus recursos de comercio virtual ofrecen la creación de comunidades de usuarios que aporten contenidos de manera que se establezca un diálogo personalizado con los clientes, quienes podrán ser así más fieles. Eso es lo que a la larga, buscan todas las tiendas virtuales. Conseguir clientes fieles y dispuestos a comprar.

Referencias

[1] Los Teatinos, "¿Qué es un Servlet?", http://losteatinos.com/servlets/servlet.html

[2] Mperez, “Java: Servlets”, Diseño de aplicaciones web.

[3] Juan Manuel Barrios, “Java Servlets”, (2001).

[4] O’Reilly & Associates, “Java Servlet Programming”, (2001)

[5] CHAITANYA SINGH, “Servlet API”

[6] “Set-Cookie”, MDN web docs, https://developer.mozilla.org/es/docs/Web/HTTP/Headers/Set-Cookie

[7] Ramón Saquete, “¿Qué son y cómo funcionan las cookies?”, HumanLevel (2013)