Instituto Politcnico Nacional

Escuela Superior de Cómputo

Web App Development.

Tarea 3 : Documentar Línea a Línea ClaseSimple.java

Profesor: M. en C. José Asunción Enríquez Zárate

Alumno: Mauro Sampayo Hernández

mauro_luigi@hotmail.com

3CM18

29 de octubre de 2021

Índice

1. Introducción 1

2. Desarrollo 2

3. Conclusión 4

4. Referencias Bibliográficas 5

1. Introducción

Por lo general cuando se realiza la programación de aplicaciones Web por medio del uso de Servlets, resulta necesario el uso de sesiones de usuario, las cuales representan una colección de objetos que un cliente especifico utiliza durante el uso de dicha aplicación Web. De esto surge la necesidad de poder administrar cada una de las sesiones de usuario que sean creadas dentro de la aplicación.

En el desarrollo de este documento se mostrará una clase de ejemplo que contiene 3 métodos que hacen uso del objeto HttpSession, que se encarga de la administración de sesiones de usuario dentro de un Servlet, así como también de métodos y algoritmos relacionados a dicho objeto que resultan muy útiles al momento de realizar tareas tales como la de crear e inicializar sesiones de usuario, y comprobar, eliminar y acceder a los valores contenidos dentro de los atributos almacenados en dichas sesiones.

2. Desarrollo

A continuacón se mostrará el código de la clase “ClaseSimple.java” propuesto por el profesor, debidamente documentado:

```
1     // Define una clase publica llamada ''ClaseSimple''
2     public class ClaseSimple{
3
4         /*  Define el metodo ''metodoUno  que recibe como parametros un request (datos
5             encapsulados de la solicitud de un cliente) y un response (datos encapsulados
6             de respuesta a la solicitud del cliente).   */
7         /*  Este metodo realiza la creacion o recuperacion de una sesion de usuario
8             y le asigna valores */
9         public void metodoUno(HttpServletRequest request, HttpServletResponse response){
10            /*  Se inicializa un objeto HttpSession para poder almacenar la informacion de
11                la sesion de usuario que va a controlar el servlet, y se inicializa con la
12                recuperacion de la sesion del request que se reciba por parte del usuario,
13                y en caso de que esta no exista se creara   */
14            HttpSession session = request.getSession(true);
15            /*  Se realiza el almacenamiento de los atributos ''nombre'' y ''valor'' dentro
16                de la sesi n actual */
17            session.setAttribute("nombre", "valor");
18        } // Fin de ''metodoUno''
19
20        /*  Define el metodo ''metodoDos'' que recibe como parametros un request (datos
21            encapsulados de la solicitud de un cliente) y un response (datos encapsulados
22            de respuesta a la solicitud del cliente).   */
23        /*  Este metodo realiza lla eliminacion del atributo ''nombre'' dentro de una
24            sesion, y en caso de que esta falle se eliminara la sesion. */
25        public void metodoDos(HttpServletRequest request, HttpServletResponse response){
26            /*  Se inicializa un objeto HttpSession para poder almacenar la informacion de
27                la sesion de usuario que va a controlar el servlet, y se inicializa con la
28                recuperacion de la sesion del request que se reciba por parte del usuario,
29                y en caso de que esta no exista se creara   */
30            HttpSession session = request.getSession(true);
31            //  Se elimina el atributo ''nombre'' dentro de la sesion de usuario actual
32            session.removeAttribute("nombre");
33            /*  Se utiliza el metodo getAttribute para obtener la informacion guardada en el
34                atributo  nombre  de la sesion actual, y en caso de que el valor devuelto
35                por el metodo no sea igual a nulo (lo cual significa que el metodo utilizado
36                para la eliminacion el atributo ''nombre'' en la linea anterior fallo de
37                alguna manera):    */
38            if(session.getAttribute("nombre") != null){
39                //  Se elimina la sesion en su totalidad
40                session.invalidate
41            } // Fin if
42        } // Fin de ''metodoDos''
43
44        /*  Define el metodo ''metodoTres'' que recibe como parametros un request (datos
45            encapsulados de la solicitud de un cliente) y un response (datos encapsulados
46            de respuesta a la solicitud del cliente).   */
47        /*  Este metodo comprueba si una sesion de usuario existe o no, y en caso de que
48            exista comprueba si cuenta con el atributo ''nombre'' */
49        public boolean metodoTres(HttpServletRequest request, HttpServletResponse response){
50            /*  Se inicializa un objeto HttpSession para poder almacenar la informacion de
51                la sesion de usuario que va a controlar el servlet, y se inicializa con la
52                recuperacion de la sesion del request que se reciba por parte del usuario,
53                y en caso de que esta no exista el valor de la sesion sera ''null''   */
54            HttpSession session = request.getSession(false);
55            // En caso de que la sesion sea igual a ''null'':
56            if(session == null){
57                // Se retorna un valor booleano ''FALSE''
58                return false;
59            // En caso contrario (el valor de la sesion es igual a algo diferente de ''null''):
60            } else {
```

```
61                  /*  Se retorna el valor booleano resultante de comprobar que el atributo
62                      ''nombre'' dentro de la sesion actual exista. En caso de que exista
63                      retornara el valor booleano ''TRUE'', y en caso contrario retornara
64                      el valor booleano ''FALSE''.  */
65                  return session.getAttribute("nombre") != null;
66              } // Fin if-else
67          } // Fin de ''metodoTres''
68      }
```

3. Conclusión

Los métodos y algoritmos presentados en el código propuesto por el profesor nos demuestran la importancia de llevar a cabo la administración de sesiones de usuario en aplicaciones Web, para que de esta manera cada request realizado por cada uno de los usuarios que estén haciendo uso de la aplicación sea administrado de manera correcta por esta y pueda crear respuestas acordes con cada solicitud que reciba.

Así mismo, a través de este ejercicio se llegó a la conclusión de que uno de los errores más típicos que cometen los programadores durante el desarrollo de aplicaciones Web es, que se suele obviar el funcionamiento de cada una de las líneas de código que conforman a la aplicación, cosa que puede llegar a afectar a largo plazo el desarrollo de la aplicación, pues alguna línea de código cuyo funcionamiento no entendamos del todo podría hacer que la aplicación en cuestión no realice los procedimientos que se quieran realizar durante la ejecución de algún método de esta, de manera correcta.

De la misma manera, y ligado a lo anteriormente mencionado, otro error común es el de no investigar acerca del funcionamiento de los métodos que se quieren implementar, lo cual puede conllevar a que el comportamiento de la aplicación que se esté desarrollando sea erróneo, pues tomando como ejemplo el método "getSession" ; si no se conoce la diferencia entre que hace el método cuando recibe como parámetro TRUE y cuando recibe como parámetro FALSE, se podrían generar errores al momento de inicializar una sesión de usuario, pues en caso de que está no exista al momento de realizar su inicialización y por consiguiente se quiera realizar la creación de la misma para su correcta inicialización, poner el parámetro incorrecto (que para este caso sería FALSE) resultaría en que esto último nunca sucediese y por lo tanto la aplicación se comporte de manera incorrecta.

Mauro Sampayo Hernández

4. Referencias Bibliográficas

Referencias

[1] Chaitanya Singh. *HttpSession with example in Servlet* **BegginersBook** [accesed 2021 Oct 27] *https://beginnersbook.com/2013/05/http-session/*

[2] *Difference Between request.getSession() and request.getSession(true)* **Baeldung, 2020** [accesed 2021 Sep 19] *https://www.baeldung.com/java-request-getsession*

[3] *Element.getAttribute()* **MDN Web Docs** [accesed 2021 Sep 19] *https://developer.mozilla.org/es/docs/Web/API/Element/getAttribute*

[4] *Servlets (Básico)* **Programacion.net** [accesed 2021 Sep 19] *https://programacion.net/articulo/servlets_basico_108/6*

[5] *Java Platform, Standard Edition 7 API Specification* **Oracle** [accesed 2021 Sep 19] *https://docs.oracle.com/javase/7/docs/api/*