educación IT
CODE YOUR FUTURE

Java Web API

Módulo 1

Sesiones

Administración de sesiones

¿Qué es una sesión?

Permite **identificar a un cliente en particular del lado del servidor**. Puede mantener un conjunto de valores determinados únicamente para ese cliente a lo largo de un periodo de tiempo.

Técnicamente, una sesión es un **espacio de memoria que reserva el `servlet` container para un cliente**. **Lo identifica con un identificador de sesión o `sessionID`**.

Las sesiones **se utilizan típicamente en *logins* de usuarios,** sitios de **e-commerce** donde resulta necesario construir un carrito de compras, y en bancos que proveen el servicio de ***Home Banking***.

El sessionID

Sirve para identificar unívocamente a un cliente. Su creación y destrucción están a cargo del `servlet` container. Es una **cadena de caracteres** sumamente larga que **combina números y letras e identifica unívocamente a un cliente** dentro del "universo" de clientes.

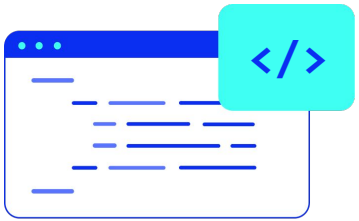
Por su parte, el protocolo HTTP es "*stateless*" o sin estado, entonces para que el servidor pueda determinar quién es el cliente, necesita obtener algún identificador.

El **identificador es precisamente el `sessionID`**, que viaja permanentemente entre el cliente y servidor de forma transparente para usuarios finales y también para desarrolladores. Gracias al sessionID, el servidor podrá brindarle al cliente un "trato personalizado".

El objeto session

Es uno de los **objetos implícitos instanciados por el Web container en tiempo de ejecución**, y modela la sesión de un usuario particular. En este objeto es posible almacenar información para luego ser recuperada. Técnicamente el **objeto `session` es una instancia de una clase que implementa la interfaz `HttpSession`**.

Es posible almacenar los valores que resulten necesarios utilizando el método **`setAttribute()`**.



La forma de **almacenar un atributo** es:

```
[code]
session.setAttribute(
“nombre_del_atributo”, objeto_con_el_valor);
[/code]
```

Por ejemplo, se podría guardar un objeto del tipo Usuario como sesión:

```
[code]
      session.setAttribute(“user”, elUsuario);
[/code]
```

Para obtener un **valor de sesión**, es necesario utiliza el método **`getAttribute()`**:

```
[code]
Usuario unUsuario =
(Usuario) session.getAttribute(“user”);
```

Session timeout

Determina el tiempo que dura la sesión sin que el usuario interactúe con el servidor.

El *Web container* controla el último pedido de un cliente con el servidor y luego de cumplido este tiempo sin interacción por parte del usuario, destruye la sesión.

Para establecer el **`session-timeout`** es necesario agregar las siguientes líneas dentro del descriptor de despliegue:

```
[code]
<session-config>
      <session-timeout>10</session-timeout>
</session-config>
[/code]
```

El tiempo del **`timeout`** se expresa en **minutos**.

Destrucción de una sesión

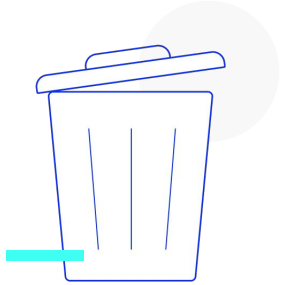
Una vez terminado el uso de los valores de sesión, la sesión debe ser destruida.

Anteriormente vimos un mecanismo de destrucción por tiempo, el `session-timeout`.

Para **eliminar una sesión de forma manual** se utiliza el método: **`invalidate()`**

El modo de utilización es:

```
session.invalidate();
```



URL rewriting

A veces, resulta necesario reescribir la URL con información adicional **cuando el browser trabaja sin las cookies**. En caso de que el browser NO acepte cookies, o estén deshabilitadas, **es necesario modificar todas las URLs para que transporten el identificador de la sesión**.

Un ejemplo de envío del identificador de sesión es:

```
../destino.jsp;jsessionid=aa363bcaa3af23bac8f4999
```

Para trabajar de **forma automática** es necesario modificar manualmente todos las URLs que formen parte de la aplicación, tanto las pertenecientes a redirecciones como las de links. Esto es muy costoso y generalmente, cuando no se aceptan cookies, no se le deja continuar navegando al usuario.

La forma de utilizarlo es la siguiente:

```
[code]
<a href=<%=response.encodeUrl(“destino.jsp”)%>
>
Destino del link
</a>
[/code]
```

El método **`response.encodeRedirectUrl()`** se utiliza en todas las redirecciones realizadas en los archivos **`jsp`**. La forma de utilizarlo es la siguiente:

```
[code]
<%
response.sendRedirect(
response.encodeRedirectUrl(“destino.jsp”) );
%>
[/code]
```

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!