DESARROLLO DE APLICACIONES WEB

Tema 5.- Java Entreprise Edition

Servlets: Filters y Listeners

Contenido

- Filters
- Web Listeners

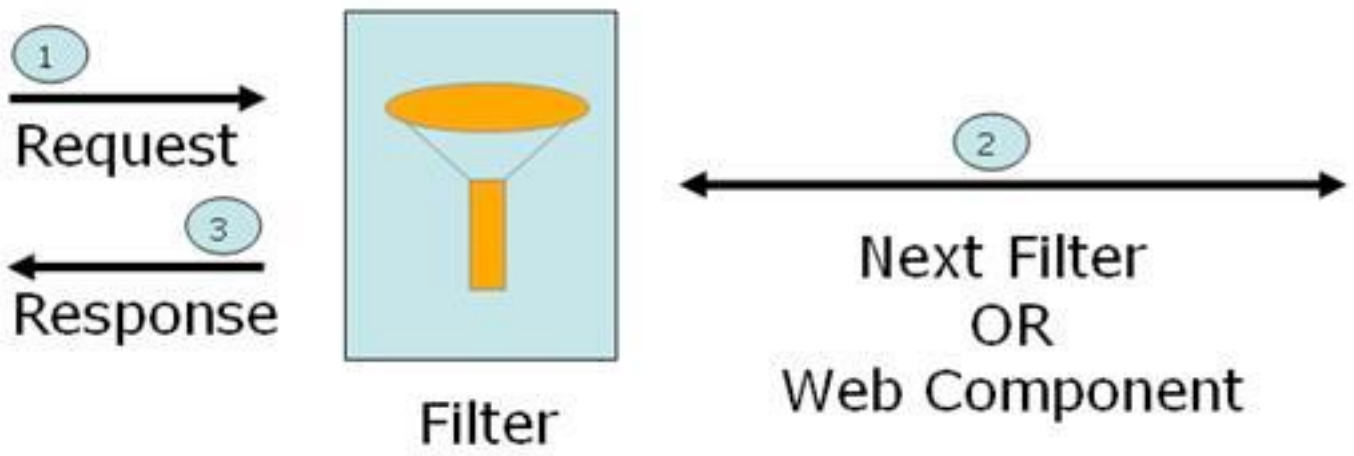
Filters

- Los filtros son clases Java cuya funcionalidad es:
  - Interceptar la petición del cliente antes de acceder al recurso solicitado
  - Interceptar la respuesta del servidor antes de que sea enviada al cliente

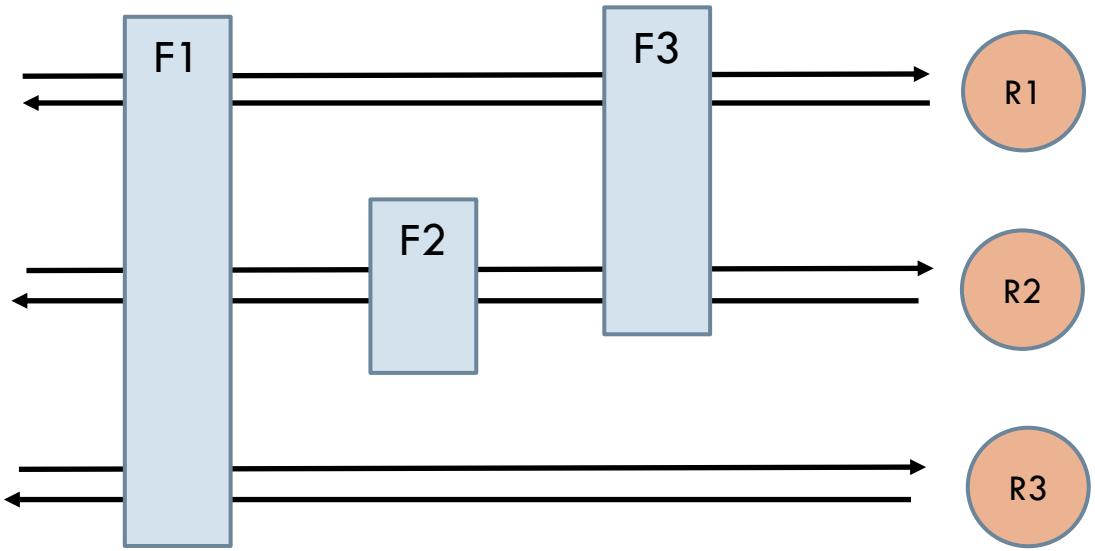
Filters

- Propósitos:
  - Autenticación y Autorización
  - Auditoria
  - Encriptación
  - Compresión de Datos
  - Conversión de Imágenes
  - etc.

Filters



Filters



Filters

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/filter

javax.servlet

Interface Filter

All Methods	Instance Methods	Abstract Methods

Modifier and Type	Method and Description
`void`	**`destroy()`** Called by the web container to indicate to a filter that it is being taken out of service.
`void`	**`doFilter(ServletRequest request, ServletResponse response, FilterChain chain)`** The `doFilter` method of the Filter is called by the container each time a request/response pair is passed through the chain due to a client request for a resource at the end of the chain.
`void`	**`init(FilterConfig filterConfig)`** Called by the web container to indicate to a filter that it is being placed into service.

Filters

```
import jakarta.servlet.Filter;
import jakarta.servlet.annotation.WebFilter;

@WebFilter(filterName = ”nombreFiltro”, urlPatterns = {"/*”} )
public class nombre implements Filter {

    // doFilter

}
```

Filters

```
@WebFilter(filterName = "Filtro1", urlPatterns = {"/Servlet1"})
public class Filtro1 implements Filter {
…
        public void doFilter(ServletRequest request,
                ServletResponse response,
                FilterChain chain)
                        throws IOException, ServletException {

            // Preprocesado

            // chain.doFilter(request, response);

            // Postprocesado

        }
…
}
```

Filters – web.xml

```
 <filter>
      <filter-name>nombreFiltro</filter-name>
      <filter-class>ruta.paquete.claseFiltro</filter-class>
</filter>


<filter-mapping>
       <filter-name>nombreFiltro</filter-name>
       <url-pattern>/*</url-pattern>
</filter-mapping>
```

Filters – Ejemplo - log

```
public void doFilter(ServletRequest req, ServletResponse res,
        FilterChain chain) throws IOException, ServletException {

    HttpServletRequest request = (HttpServletRequest) req;

    // IP máquina cliente.
    String ipAddress = request.getRemoteAddr();

    // Muestra hora Petición.
    System.out.println("IP "+ipAddress + ", Time “ + new Date().toString());

    chain.doFilter(req, res);

    // Muestra hora Respuesta.
    System.out.println("IP "+ipAddress + ", Time “ + new Date().toString());

}
```

Filters – Ejemplo - Login

```
 public void doFilter(ServletRequest req, ServletResponse res, FilterChain
chain) throws ServletException, IOException {

      HttpServletRequest request = (HttpServletRequest) req;
      HttpSession session = request.getSession();

      if (session.getAttribute(“user”) ==null) {
            // usuario no identificado
            req.getRequestDispatcher(“login.jsp”).forward(req, res);
      } else {
          chain.doFilter(req, res);
      }
  }
```

Contenido

- Filters
- Web Listeners

Listeners

- Monitorizan y actúan frente a eventos durante el ciclo de vida de un servlet

- Básicamente, en los ámbitos:

  - Global o Aplicación (ServletContext)
  - Sesión (HttpSession)
  - Petición (ServletRequest)

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/annotation/weblistener

Listeners - Ejemplos

```
@WebListener
public class nombrelistener  implements interfaceListener {
…
}
```

Interfaces

- ServletContextListener
- ServletContextAttributeListener
- HttpSessionListener
- HttpSessionAttributeListener
- ServletRequestListener
- ServletRequestAttributeListener

Listeners

- Fichero web.xml

```
<listener>
        <description>Descripción</description>
        <listener-class>ClaseListener</listener-class>
 </listener>
```

- Anotaciones

```
@WebListener
public class ClaseListener implements InterfaceImplementar { …
```

Listeners

- Ámbito Application:
  - ServletContextListener: Interface que recibe notificaciones cuando se producen cambios en el ciclo de vida del contexto de la aplicación.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletcontextlistener

  - ServletContextAttributeListener: Interface que recibe notificaciones cuando añadimos atributos al contexto (ámbito global), los eliminamos, o los reemplazamos.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletcontextattributelistener

Listeners

- Ámbito Session:
  - HttpSessionListener: Interface que recibe notificaciones cuando se crea o se destruye la Session.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/http/httpsessionlistener
  - HttpSessionAttributeListener: Recibe notificaciones cuando añadimos atributos a la Session, los eliminamos o los reemplazamos.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/http/httpsessionattributelistener

Listeners

- Ámbito Petición (Servlet)
  - ServletRequestListener, Interface que recibe notificaciones en cada petición que hacemos a nuestra aplicación.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletrequestlistener
  - ServletRequestAttributeListener: Interface que recibe notificaciones cuando se añaden, eliminan o actualizan objetos en el ámbito de la petición.

    https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletrequestattributelistener

Listeners: Ejemplos

- Ejemplo1. Monitorizar la creación de atributos globales
  - Interface: ServletContextAttributeListener

void	**attributeAdded**(**ServletContextAttributeEvent** event) Recibe notificaciones cuando un atributo se añade al ServletContext.
void	**attributeRemoved**(**ServletContextAttributeEvent** event) Recibe notificaciones cuando un atributo se elimina al ServletContext.
void	**attributeReplaced**(**ServletContextAttributeEvent** event) Recibe notificaciones cuando cambia un atributo del ServletContext.

  - Parámetro: ServletContextAttributeEvent

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletcontextattributeevent

String	**getName**() Devuelve el nombre del atributo implicado.
Object	**getValue**() Devuelve el valor del atributo implicado..

Listener - Ejemplos

```
@WebListener
public class AppContextAttributeListener implements ServletContextAttributeListener {
    public void attributeAdded(ServletContextAttributeEvent sCAEvent) {
        System.out.println(”Añadido ::{"+sCAEvent.getName()+","+
                                                    +sCAEvent.getValue()+"}");
    }
    public void attributeReplaced(ServletContextAttributeEvent sCAEvent) {
        System.out.println(”Reemplazado::{"+sCAEvent.getName()+","
                                                     +sCAEvent.getValue()+"}");
    }
    public void attributeRemoved(ServletContextAttributeEvent sCAEvent) {
        System.out.println(”Borrado::{"+sCAEvent.getName()+","+sCAEventgetValue()+"}");
    }
}
```

Listeners: Ejemplos

- Ejemplo 2. Monitorizar la sesión
  - Interface: HttpSessionListener

void	**sessionCreated**(**HttpSessionEvent** se) Recibe notificaciones cuando se crea la sesión
void	**sessionDestroyed**(**HttpSessionEvent** se) Recibe notificaciones cuando se invalida la sesión

  - Parámetro: HttpSessionEvent

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/http/httpsessionevent

HttpSession	**getSession**() Devuelve la sesión que ha cambiado

Listener - Ejemplos

```
@WebListener
public class MySessionListener implements HttpSessionListener {

    public void sessionCreated(HttpSessionEvent sessionEvent) {
        System.out.println("Session Created:: ID="+sessionEvent.getSession().getId());
    }

    public void sessionDestroyed(HttpSessionEvent sessionEvent) {
        System.out.println("Session Destroyed:: ID="+sessionEvent.getSession().getId());
    }

}
```

Listeners: Ejemplos

- Ejemplo 3. Monitorizar los atributos de la sesión
  - Interface: HttpSessionListener

void	**attributeAdded**(**HttpSessionBindingEvent** event) Recibe notificaciones cuando un atributo es añadido a la sesión.
void	**attributeRemoved**(**HttpSessionBindingEvent** event) Recibe notificaciones cuando un atributo es eliminado de la sesión.
void	**attributeReplaced**(**HttpSessionBindingEvent** event) Recibe notificaciones cuando se modifica un atributo de la sesión.

  - Parámetro: HttpSessionBindingEvent

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/http/httpsessionbindingevent

String	**getName**() Devuelve el nombre del atributo implicado.
Object	**getValue**() Devuelve el valor del atributo implicado.

Listener - Ejemplos

```
public class SessionAttListener implements HttpSessionAttributeListener {


    @Override
    public void attributeAdded(HttpSessionBindingEvent event) {
        System.out.println("Añadido Att " + event.getName() + " Sesión " + event.getSession().getId());
    }
    @Override
    public void attributeRemoved(HttpSessionBindingEvent event) {
        System.out.println("Eliminado Att de Sesión: " + event.getSession().getId());
    }
    @Override
    public void attributeReplaced(HttpSessionBindingEvent event) {
        System.out.println("Modificado Att en Sesión: " + event.getSession().getId());
    }
}
```

Listeners: Ejemplos

- Ejemplo 4. Monitorizar las peticiones
  - Interface: ServletRequestListener

void	**requestDestroyed**(**ServletRequestEvent** sre) Recibe notificaciones cuando finaliza una petición a un Servlet.
void	**requestInitialized**(**ServletRequestEvent** sre) Recibe notificaciones cuando se realiza una petición a un Servlet.

  - Parámetro: ServletRequestEvent

https://jakarta.ee/specifications/platform/10/apidocs/jakarta/servlet/servletrequestevent

ServletContext	**getServletContext**() Devuelve el context del Servlet (ServletContext)
ServletRequest	**getServletRequest**() Devuelve la petición implicada.

Listener - Ejemplos

```
@WebListener
public class MyServletRequestListener implements ServletRequestListener {


    public void requestDestroyed(ServletRequestEvent sREvent) {
        ServletRequest servletRequest = sREvent.getServletRequest();
        System.out.println(”Petición Finalizada. IP="+servletRequest.getRemoteAddr());
    }
    public void requestInitialized(ServletRequestEvent sREvent) {
        ServletRequest servletRequest = sREvent.getServletRequest();
        System.out.println(”Petición Iniciada. IP="+servletRequest.getRemoteAddr());
    }
}
```