MÓDULO 5

EJERCICIOS RESUELTOS FILTROS

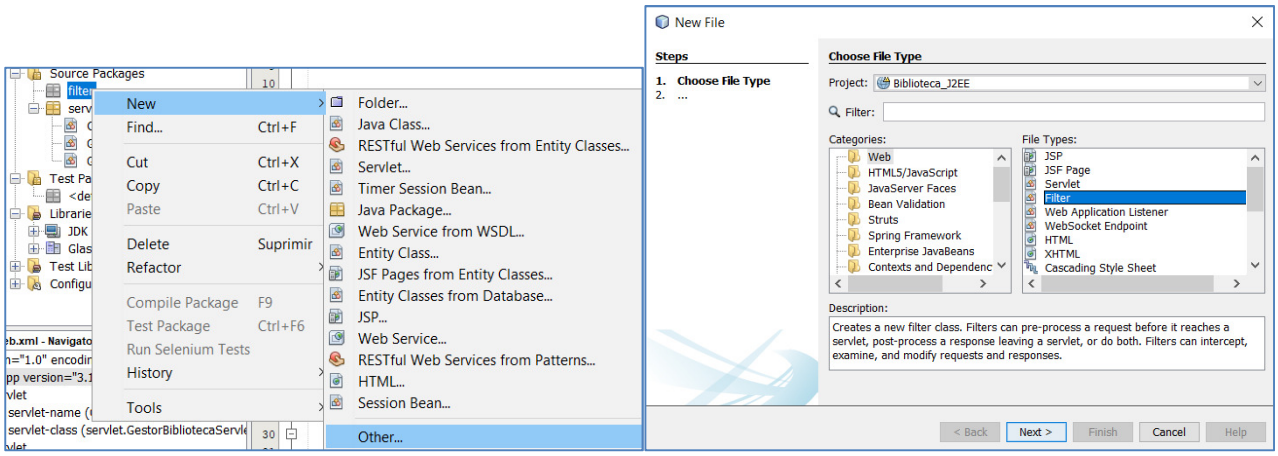
TEORIA 1

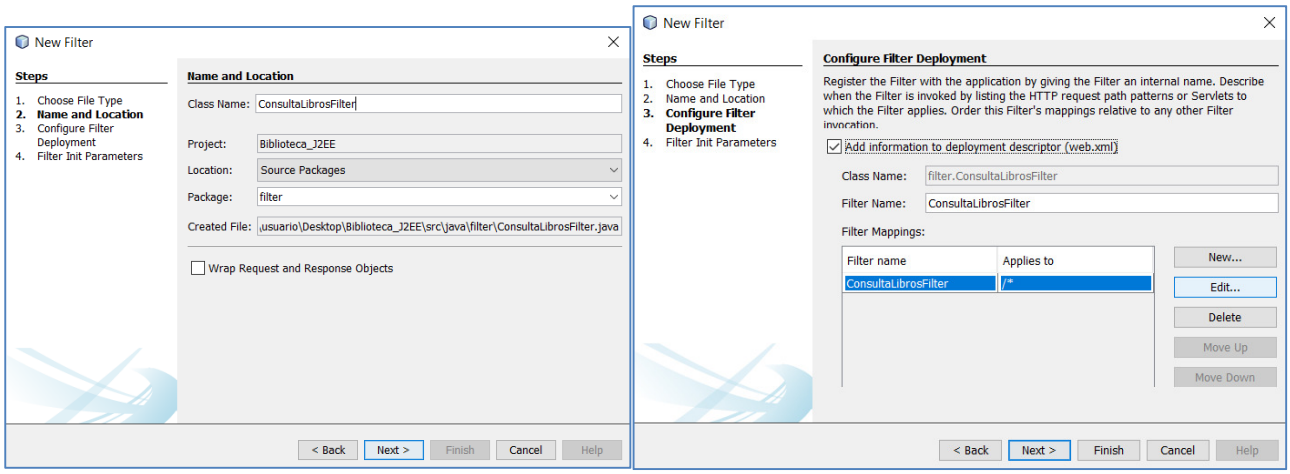
Explica que es un filtro, y qué relación tiene con los servlets.

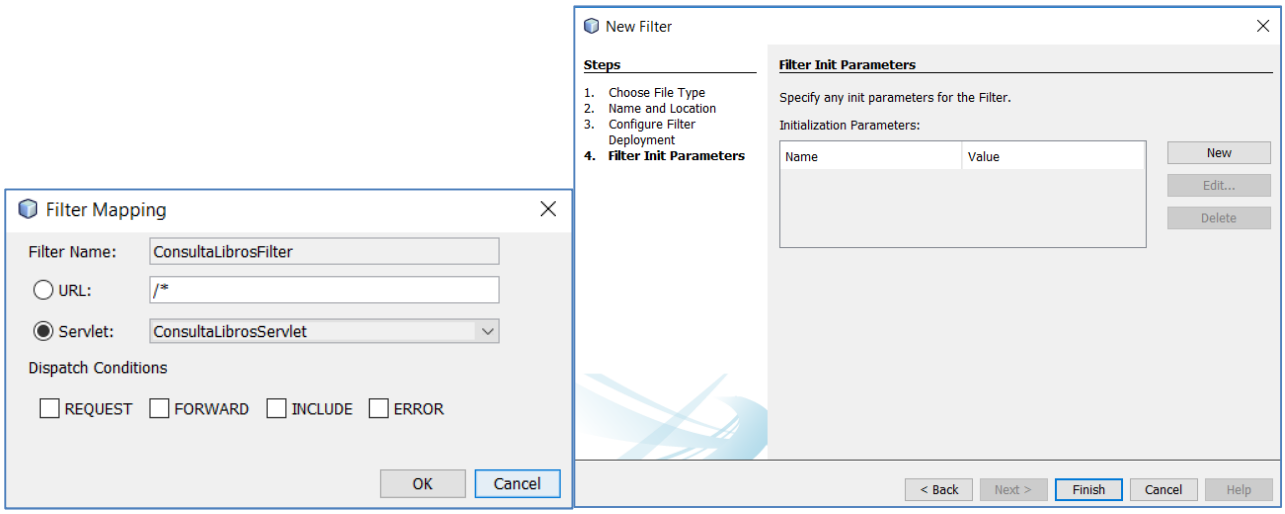
EJERCICIO 1

Crea la plantilla de un filtro llamado ConsultaLibrosFilter en el package filter, que deberemos asignar al servlet ConsultaLibrosServlet. Debe contener los métodos init, destroy, doFilter y toString, implementados por la interficie Filter.

NOTA: En los siguientes apartados haremos que este filtro registre en un fichero de texto todas las consultas que realizan los usuarios, además de su IP. La idea es realizar estadísticas sobre el uso de nuestra web.








```java
public class ConsultaLibrosFilter implements Filter {

    private FilterConfig filterConfig = null;

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
    }

    private void doBeforeProcessing(ServletRequest request, ServletResponse response)
            throws IOException, ServletException {
    }

    private void doAfterProcessing(ServletRequest request, ServletResponse response)
            throws IOException, ServletException {
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain)
            throws IOException, ServletException {

        doBeforeProcessing(request, response);
        chain.doFilter(request, response);
        doAfterProcessing(request, response);
    }

    @Override
    public void destroy() {
    }

    @Override
    public String toString() {
        return ("");
    }
}
```

EJERCICIO 2

Indica las reglas necesarias para definir el filtro ConsultaLibrosFilter en web.xml y su asociación al servlet ConsultaLibrosServlet.

```
<filter>
    <filter-name>ConsultaLibrosFilter</filter-name>
    <filter-class>filter.ConsultaLibrosFilter</filter-class>
</filter>
<filter-mapping>
    <filter-name>ConsultaLibrosFilter</filter-name>
    <servlet-name>ConsultaLibrosServlet</servlet-name>
</filter-mapping>
```

EJERCICIO 3

Configura un parámetro en el fichero web.xml para que lo pueda leer el filtro ConsultaLibrosFilter, que indique el path del fichero donde se almacenarán los registros de consulta.

Realiza la lectura de dicho parámetro en la función init del filtro ConsultaLibrosFilter.

```
<filter>
    <filter-name>ConsultaLibrosFilter</filter-name>
    <filter-class>filter.ConsultaLibrosFilter</filter-class>
    <init-param>
        <param-name>Registro</param-name>
        <param-value>C:\\Registro.txt</param-value>
    </init-param>
</filter>

<filter-mapping>
    <filter-name>ConsultaLibrosFilter</filter-name>
    <servlet-name>ConsultaLibrosServlet</servlet-name>
</filter-mapping>
```

```
public class ConsultaLibrosFilter implements Filter {

    private FilterConfig filterConfig = null;
    private String ruta;

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
        ruta = filterConfig.getInitParameter("Registro");
    }
```

EJERCICIO 4

Realiza la función pre-filtro del filtro ConsultaLibrosFilter. Debe registrar en un fichero de texto todas las consultas que realizan los usuarios, además de su IP. La idea es realizar estadísticas

sobre el uso de nuestra web. El formato debe ser:

Usuario: xxxxx – IP:xxxxxxx – Titulo:xxx

¿Tiene acceso el filtro a todos los parámetros a registrar (consulta, usuario e IP)?

```
public class ConsultaLibrosFilter implements Filter {
    private FilterConfig filterConfig = null;
    private String ruta;

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
        ruta = filterConfig.getInitParameter("Registro");
    }

    private void doBeforeProcessing(ServletRequest request, ServletResponse response)
            throws IOException, ServletException {
        InetAddress address = InetAddress.getLocalHost();
        String sHostName = address.getHostName();
        //String usuario = request.getAttribute("usuario").toString();
        String usuario = request.getParameter("usuario");
        String titulo = request.getParameter("titulo");
        String linea = "Usuario: " + usuario + " IP: " + address.toString() + " Titulo: " + titulo  + "\n";
        File fichero = new File(ruta);
        if (fichero.exists()) {
            BufferedWriter bw = new BufferedWriter(new FileWriter(fichero,true));
            bw.write(linea);
            bw.close();
        }else{
            fichero.createNewFile();
            BufferedWriter bw = new BufferedWriter(new FileWriter(fichero));
            bw.write(linea);
            bw.close();
        }
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain)
            throws IOException, ServletException {

        doBeforeProcessing(request, response);
        chain.doFilter(request, response);
        doAfterProcessing(request, response);
    }
```

La dirección IP se puede conseguir. El título de la consulta también, puesto que ya viene en el formulario, pero el nombre del usuario no se tiene acceso, ya que venimos del fichero bienvenido.jsp y ese parámetro no está incluido.

EJERCICIO 5

La consulta se realiza desde el formulario de bienvenido.jsp, y el filtro se ejecuta antes de entrar en el servlet ConsultaLibrosServlet (pre-filtro). Como no se pueden aplicar sesiones, indica los cambios necesarios en bienvenido.jsp para que el filtro pueda obtener correctamente todos los parámetros.

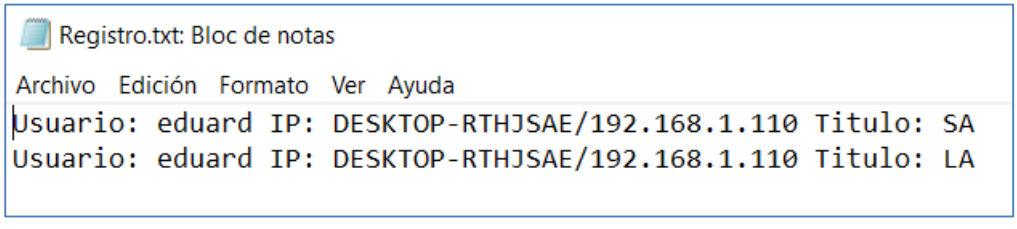

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>BIENVENIDA</title>
    </head>
    <body>
            <%String usuario=request.getParameter("usuario");%>
            <%String iniciado=request.getParameter("iniciado");%>

            <%if (iniciado.equals("false")){%>
                <h1>Conectado a la BD (llamada GET)</h1>
            <%}%>

            <h1>BIENVENIDO USUARIO <%=usuario%></h1>
            <form method=GET action=ConsultaLibrosServlet>
                Selección de Libro: <input type='text' name='titulo'><br><br>
                <input type=submit>
            </form>

    </body>
</html>
```

```jsp
<h1>BIENVENIDO USUARIO <%=usuario%></h1>
<form method=GET action=ConsultaLibrosServlet>
    Selección de Libro: <input type='text' name='titulo'><br><br>
    <input type='hidden' name='usuario' value='<%=usuario%>'>
    <input type=submit>
</form>
```



EJERCICIO 6

Crea un segundo filtro que lea la respuesta de ConsultaLibrosServlet, la pase a minúsculas y la escriba en un fichero. ¿Qué alternativas existen para realizar esta funcionalidad?

Como es una funcionalidad post filtro del servlet ConsultaLibrosServlet, o podemos crear un nuevo filtro o usar la función post-filtro del filtro ConsultaLibrosFilter. Implementaremos esta segunda opción.

EJERCICIO 7

Una vez elegida la solución de filtro más simple, configura un parámetro en el fichero web.xml para el nuevo filtro, que indique el path del fichero donde se almacenará la consulta.

Realiza la lectura de dicho parámetro en la función init del nuevo filtro.

```
public class ConsultaLibrosFilter implements Filter {
    private FilterConfig filterConfig = null;
    private String ruta;
    private String ruta_consulta;

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
        ruta = filterConfig.getInitParameter("Registro");
        ruta_consulta = filterConfig.getInitParameter("Consulta");
    }
```

```
<filter>
    <filter-name>ConsultaLibrosFilter</filter-name>
    <filter-class>filter.ConsultaLibrosFilter</filter-class>
    <init-param>
        <param-name>Registro</param-name>
        <param-value>C:\\Registro.txt</param-value>
    </init-param>
    <init-param>
        <param-name>Consulta</param-name>
        <param-value>C:\\Consulta.txt</param-value>
    </init-param>
</filter>
```

EJERCICIO 8

Realiza la función post-filtro del filtro a minúsculas. Debe registrar en un fichero de texto el resultado de las consultas que realizan los usuarios, pero en minúsculas. El formato debe ser:

Usuario: xxxxx Titulo: xxxxx

Consulta: xxxx

Usuario: xxxxx Titulo: xxxxx

Consulta: xxxxx

NOTA: A diferencia del pre-filtro, donde los parámetros vienen de bienvenido.jsp a través de un formulario, el postfiltro viene después del servlet ConsultaLibrosServlet. En el primer caso los parámetros son recogidos con getParameter, en el segundo con getAttribute.

```java
public class ConsultaLibrosFilter implements Filter {
    private FilterConfig filterConfig = null;
    private String ruta;
    private String ruta_consulta;

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
        ruta = filterConfig.getInitParameter("Registro");
        ruta_consulta = filterConfig.getInitParameter("Consulta");
    }

    private void doAfterProcessing(ServletRequest request, ServletResponse response)
            throws IOException, ServletException {
        String consulta = (String)request.getAttribute("consulta").toString().toLowerCase();
        String usuario = (String)request.getAttribute("usuario").toString();
        String titulo = request.getAttribute("titulo").toString();
        String linea = "Usuario: " + usuario  + " Titulo: "+titulo+"\r\nConsulta: " + consulta + "\r\n";
        File fichero = new File(ruta_consulta);
        if (fichero.exists()) {
            BufferedWriter bw = new BufferedWriter(new FileWriter(fichero,true));
            bw.write(linea);
            bw.close();
        }else{
            fichero.createNewFile();
            BufferedWriter bw = new BufferedWriter(new FileWriter(fichero));
            bw.write(linea);
            bw.close();
        }
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain)
            throws IOException, ServletException {
        doBeforeProcessing(request, response);
        chain.doFilter(request, response);
        doAfterProcessing(request, response);
    }
```

EJERCICIO 8

Realiza los cambios necesarios en el servlet ConsultaLibrosServlet para que el postfiltro recoja correctamente los parámetros.

```java
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    String titulo = request.getParameter("titulo");
    String usuario = request.getParameter("usuario");
    //String path = getServletContext().getRealPath("/");
    //File f = new File(path + "libros.txt");
    File f = new File(ruta_libros);
    BufferedReader entrada = new BufferedReader(new FileReader(f));

    PrintWriter out = response.getWriter();
    out.println("<!DOCTYPE html>");
    out.println("<html>");
    out.println("<head><title>Servlet ConsultaLibrosServlet</title></head>");
    out.println("<body>");
    StringBuffer sb = new StringBuffer();
    while (entrada.ready()){
        String linea=entrada.readLine();
        boolean presencia = linea.contains(titulo);
        if (presencia){
            out.println("<h3>"+linea+"</h3>");
            sb.append(linea+"\r\n");
        }
    }
    out.println("</body>");
    out.println("</html>");
    request.setAttribute("consulta", sb.toString());
    request.setAttribute("usuario", usuario);
    request.setAttribute("titulo", titulo);
}
```

Consulta.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
Usuario: eduard Titulo: SA
Consulta: 00000001;harry potter y el prisionero de azkabán;j.k rowins;salamandra;26/9/2006 0:00:00;infantil;
00000002;el gran laberinto;fernando sabater perez;ariel;26/9/2006 0:00:00;ficción;
00000003;romeo y julieta;william shakespeare;salamandra;26/9/2006 0:00:00;romantica;
00000004;la carta esferica;arturo perez lopez;salamandra;29/9/2006 0:00:00;ficción;
00000006;mucho ruido y pocas nueces;william shakespeare;salamandra;29/9/2006 0:00:00;romantica;
00000007;protocolo;jose lopez murillo;salamandra;6/9/2006 0:00:00;social;
00000008;linux;fernando sabater perez;ariel;6/9/2006 0:00:00;informática;
00000011;el tiempo;j.k rowins;salamandra;7/9/2006 0:00:00;ciencia;

Usuario: eduard Titulo: LA
Consulta: 00000001;harry potter y el prisionero de azkabán;j.k rowins;salamandra;26/9/2006 0:00:00;infantil;
00000002;el gran laberinto;fernando sabater perez;ariel;26/9/2006 0:00:00;ficción;
00000003;romeo y julieta;william shakespeare;salamandra;26/9/2006 0:00:00;romantica;
00000004;la carta esferica;arturo perez lopez;salamandra;29/9/2006 0:00:00;ficción;
00000006;mucho ruido y pocas nueces;william shakespeare;salamandra;29/9/2006 0:00:00;romantica;
00000007;protocolo;jose lopez murillo;salamandra;6/9/2006 0:00:00;social;
00000011;el tiempo;j.k rowins;salamandra;7/9/2006 0:00:00;ciencia;
00000015;capitan alatriste;arturo perez lopez;alfaguara;9/10/2006 0:00:00;ficción;
```

EJERCICIO 9

Hemos visto que un servlet se guarda en la memoria del servidor de aplicaciones, ya que entre sus diferentes llamadas guarda el valor de sus atributos miembros. Pero un filtro ¿tiene memoria en el servidor de aplicaciones entre sus llamadas pre-filtro y post-filtro?

La respuesta la encontramos en los apartados anteriores y es totalmente negativa. Un filtro no se

comporta como un servlet. Hemos visto que en el pre-filtro, donde los parámetros vienen de bienvenido.jsp a través de un formulario, recogíamos los parámetros con getParameter. Y en el postfiltro teníamos que volver a recoger los parámetros 'usuario' y 'titulo' mediante getAttribute enviados por el servlet ConsultaLibrosServlet, puesto que entre la primera y la segunda llamada el filtro es destruido y en la segunda está completamente vacío (los atributos valen null sino no se vuelven a capturar).

EJERCICIO 10

Realiza un esquema de la aplicación creada hasta la fecha.