Java Servlets – Filters and Listeners

Presented by

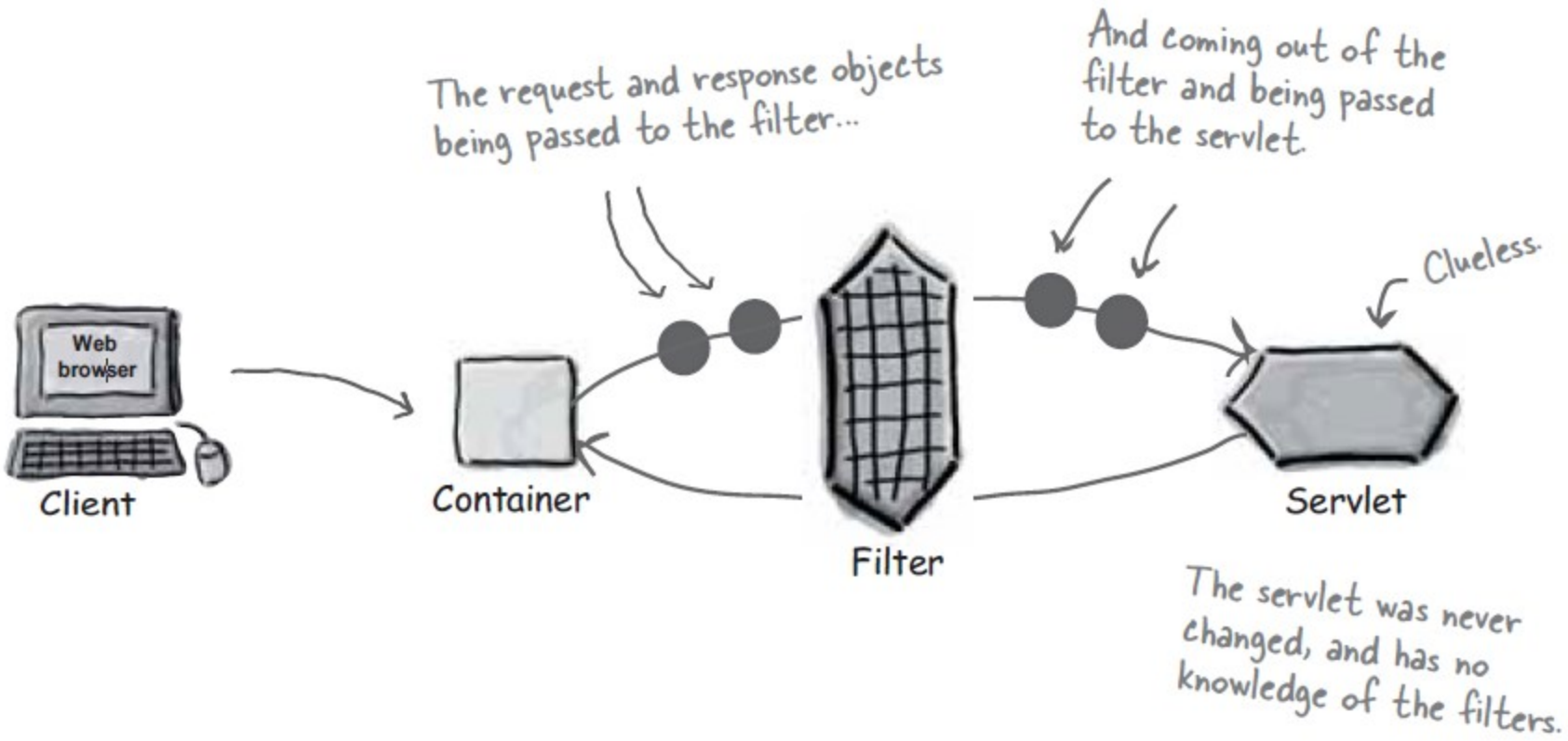
Sagar

Java Consultant

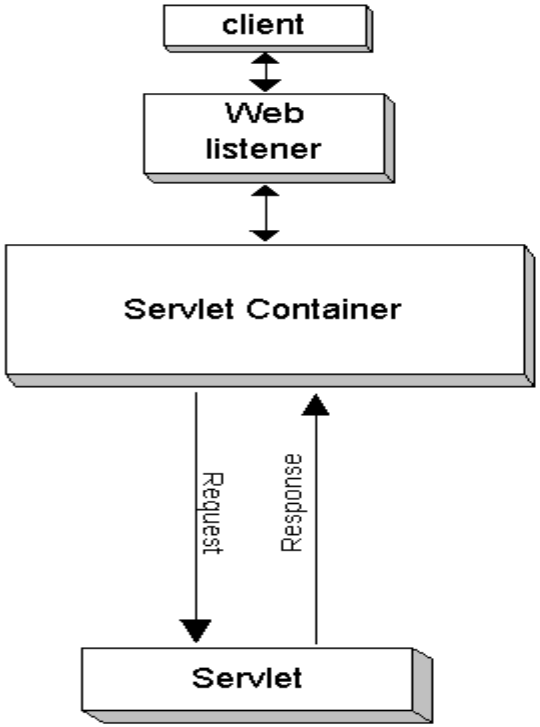
Servlet Filters

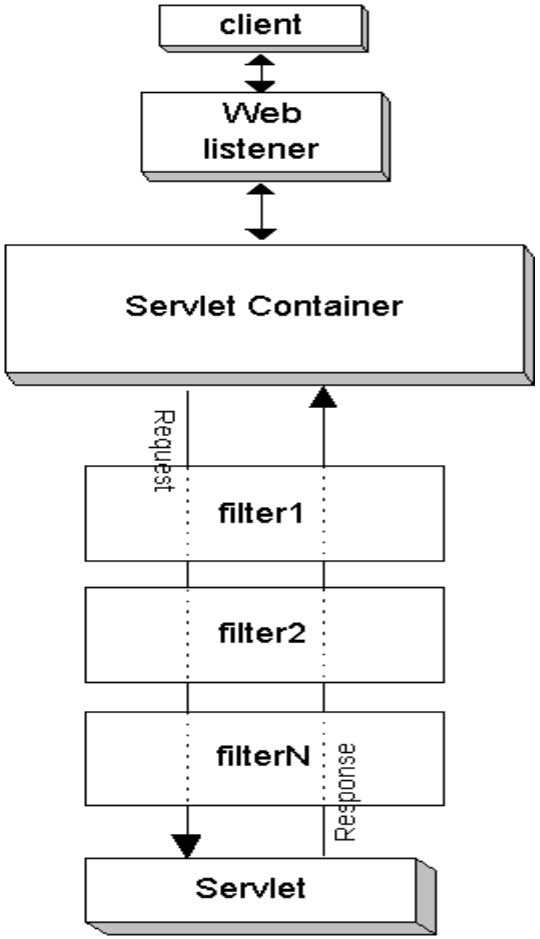
- Filters let us to intercept the request.
- Dynamic components, usually associated with servlets.
- They intercepts the requests before sending to a servlet.
- The container decides when to invoke the filters , based on the declarations
- In the Deployment Descriptor.

Servlets Filters



Servlets Filters





**Example

Example - Filter

```
public class HelloWorldFilter implements Filter{

   public void init(FilterConfig fltrCnfg) throws ServletException {}

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

      PrintWriter out=resp.getWriter();
   //Before Servlet
      out.print("I Filtered the request...!");
   // Navigates to servlet
      chain.doFilter(req, resp);//sends request to next resource, i.e, servlet
   //After servlet
      out.print("That's the power of filters..!");
   }
```

Example - Servlet

```
public class HelloWorldServlet extends HttpServlet {
   public void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {

      response.setContentType("text/html");
      PrintWriter out = response.getWriter();

      out.print("<br>Welcome to the world of Servlet Filters..!<br>");

   }

}
```

** Absolutely no clue about the filter

Deployment Descriptor

```
<filter>
    <filter-name>HelloWorldFilter</filter-name>
    <filter-class>HelloWorldFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>HelloWorldFilter</filter-name>
  <url-pattern>/HelloWorldServlet</url-pattern>
</filter-mapping>
```

** Demo

Servlet Listeners

During the lifetime of a typical web application, a number of events take place, such as:

- requests are created or destroyed.
- request or session attributes are added, removed, or modified etc.
- changing state of an object

Listeners are the classes which listens to a particular type of events.

And when that event occurs , a functionality triggers.

Each type of listener is bind to a type of event.

Servlet Listeners …

Some important tasks, such as

- counting total and current logged-in users
- creating tables of the database at the time of deploying project
- creating database connection object etc.

** Listeners are Interfaces

** Events are classes

Servlet Listeners & Events

ServletRequestListener	ServletRequestEvent
ServletContextListener	ServletContextEvent
HttpSessionListener	HttpSessionEvent

ServletRequestListener

ServletRequestListener listens to **ServletRequestEvent** which gives a notification, when request is created or destroyed.

Methods : void requestDestroyed(ServletRequestEvent e)

void requestInitialized(ServletRequestEvent e)

** Demo

ServletContextListener

ServletContextEvent class gives notifications about changes to the servlet context of a web application.

ServletContextListener receives the notifications about changes to the servlet context and perform some action.

Methods:

Methods	Description
void `contextDestroyed(ServletContextEvent e)`	is invoked when the application is destroyed.
void `contextInitialized(ServletContextEvent e)`	is invoked when the application is initialized.

** Demo

ServletSessionListener

The HttpSessionEvent is notified when session object is changed.

The corresponding Listener interface for this event is HttpSessionListener.

We can perform some operations at this event such as counting total and current logged-in users, maintaining a log of user details such as login time, logout time etc.

Methods :

When session is created:

```
public void sessionCreated(HttpSessionEvent event) { }
```

Before Session closing:

```
public void sessionDestroyed(HttpSessionEvent se) {}
```

Thank You