

# Filter

*#Filter #Controller #JavaEE*

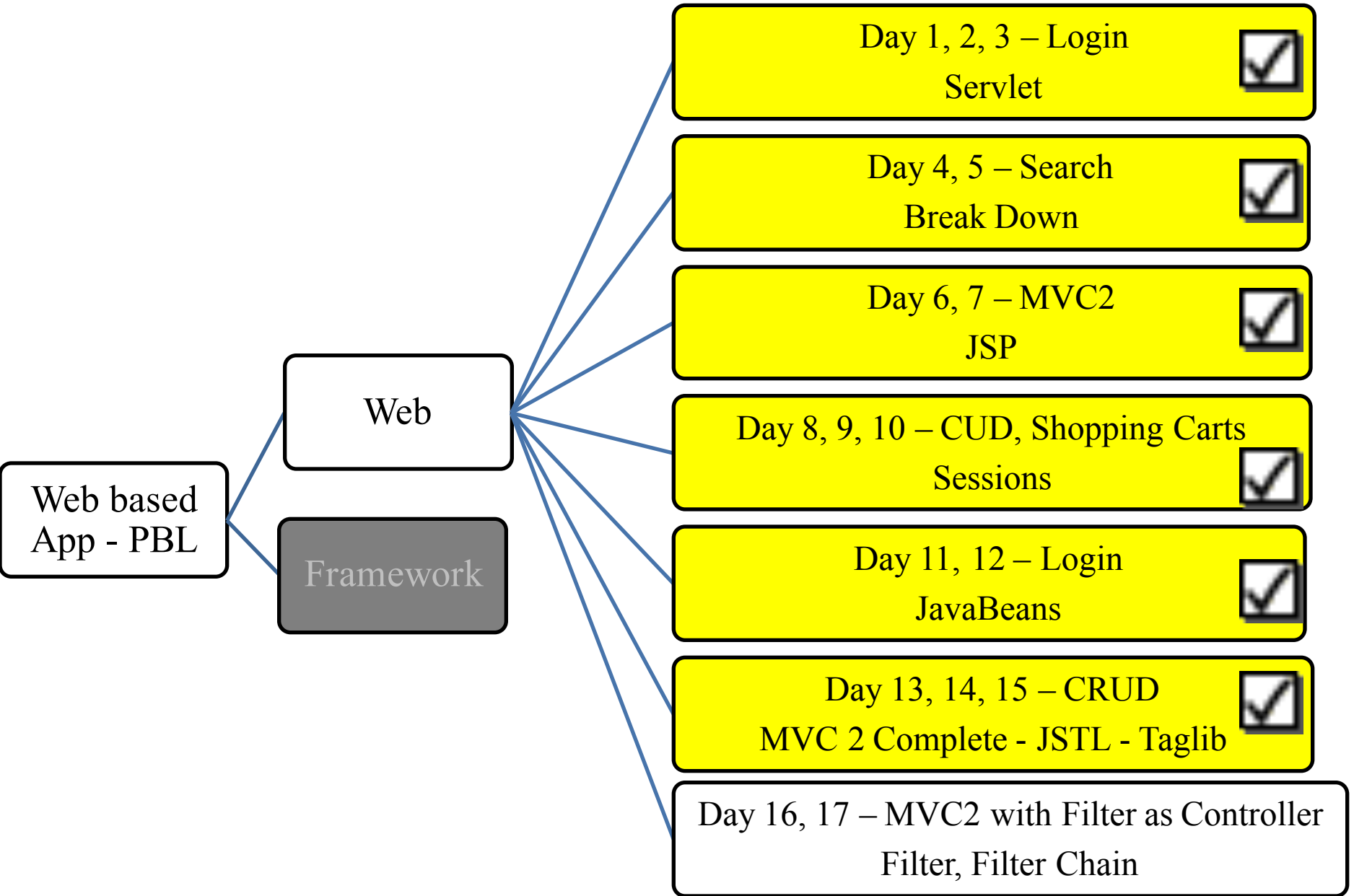
# Review

- **How to remove all java code in JSP (View)?  
Complete the MVC 2 Design Pattern with View**
  - JSTL
- **How to build the data grid tag library using in JSP?**
  - **Tag Libraries**
    - Model
    - Classical, Simple, and Handles
    - How to implement the custom Tag Lib and use it in JSP

# Objectives

- **How to build application using MVC2 Pattern using Filter as Controller?**
  - Filter
  - Filter Chain

# Objectives



# Filter

## Requirements

- **Preprocess request and Postprocess response in web application**
- **Build the Project CRUD, Shopping applying MVC2 pattern with Filter as Controller**

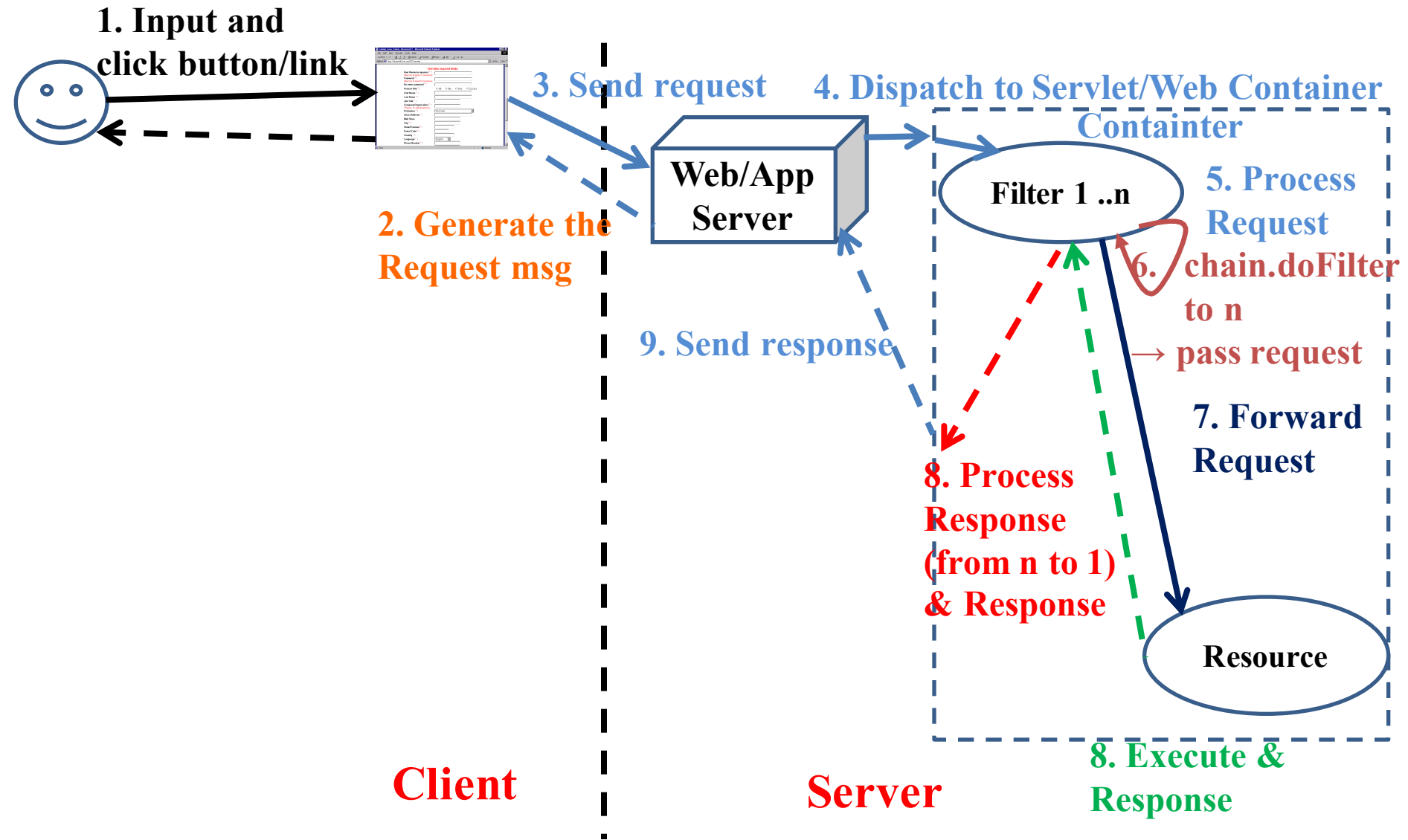
# Filter

## Overview

- **Are components that add functionality to the request and response processing of a Web Application**
  - **Intercept** the requests and response that flow between a client and a Servlet/JSP.
  - Supports **dynamic modification of requests and responses** between client and web applications.
  - Dynamically **access incoming requests** from the user before the servlet processes the request
  - **Access the outgoing response** from the web resources before it reaches the user
- **Categorized** according to the **services** they provide to the web applications
- **Resides** in the **web container** along with the web applications
- Was introduced as a Web component in Java servlet specification version 2.3

# Filter

## Interactive Filter Model



# Filter

## Usage

- Authorize request
- Altering request headers and modify data
- Modify response headers and data
- Authenticating the user
- Comprising files
- Encrypting data
- Converting images
- Logging and auditing filters
- Filters that trigger resource access events



# Filter

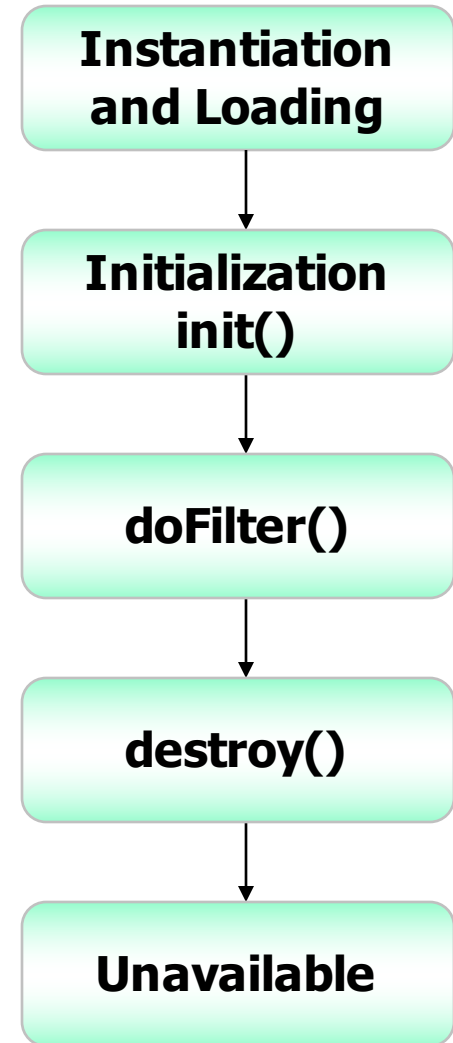
## Benefits - Advantages

- **Optimization** of the **time** taken to send a response
- **Compression** of the **content** size before sending
- **Optimization** of the **bandwidth**
- **Security**
- **Identify** the **type of request** coming from the Web client, such as HTTP and FTP, and invoke the Servlet that needs to process the request.
- **Retrieve** the user information from the request parameters to authenticate the user.
- **Validate** a client using Servlet filters before the client accesses the Servlet.
- **Identify** the **information** about the MIME types and other header contents of the request.
- **Facilitate** a **Servlet** to **communicate** with the **external resources**.
- **Intercept** responses and compress it before sending the response to the client

# Filter

## Life Cycle

- Working of Filter
  - The filter intercepts the request from a user to the servlet
  - The filter then provides customized services
  - The filter sends the serviced response or request to the appropriate destination



# Filter API

- **Creates and handles** the functionalities of a filter
- Contains **three interfaces**
  - Filter Interface, FilterConfig Interface, FilterChain Interface
- **Filter Interface**
  - Must be implemented to create a filter class **extends `javax.servlet.Filter`**
  - An object performs filtering tasks on the request and the response

Methods	Descriptions
init	<ul style="list-style-type: none"><li>- <b><code>public void init(FilterConfig fg);</code></b></li><li>- Called by the servlet container to initialize the filter</li><li>- Called only once</li><li>- Must complete successfully before the filter is asked to do any filtering work</li></ul>
doFilter	<ul style="list-style-type: none"><li>- <b><code>public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain) throws IOException, ServletException</code></b></li><li>- Called by the container each time a request or response is processed</li><li>- Then examines the request/response headers &amp; customizes them as per the requirements</li><li>- Passed the request/response through the FilterChain object to the next entity in the chain</li></ul>
destroy	<ul style="list-style-type: none"><li>- <b><code>public void destroy();</code></b></li><li>- Called by the servlet container to inform the filter that its service is no more required</li><li>- Called only once.</li></ul>

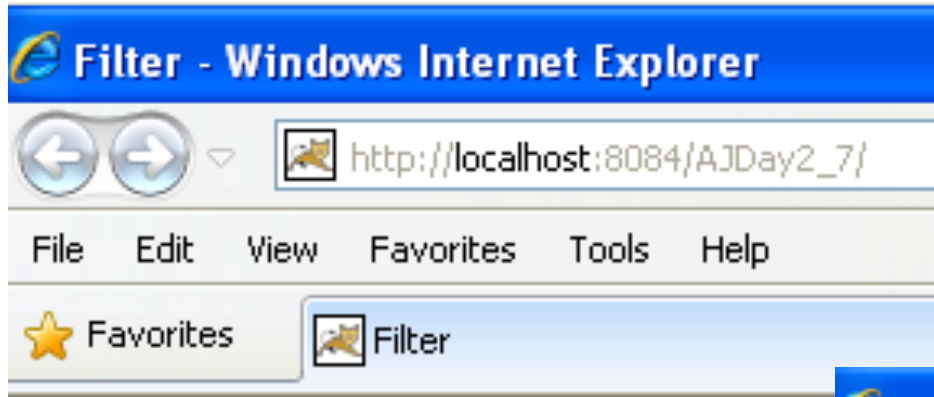
# Filter Configuration

- In Web Deployment Descriptor

```
<web-app>
....
<filter>
  <filter-name>Name of Filters</filter-name>
  <filter-class>implemented Filter Class</filter-class>
  [<init-param>
    <param-name>parameter name</param-name>
    <param-value>value </param-value>
  </init-param>]
</filter>
<filter-mapping>
  <filter-name>FilterName</filter-name>
  <url-pattern>/context</url-pattern>
</filter-mapping>
....
</web-app>
```

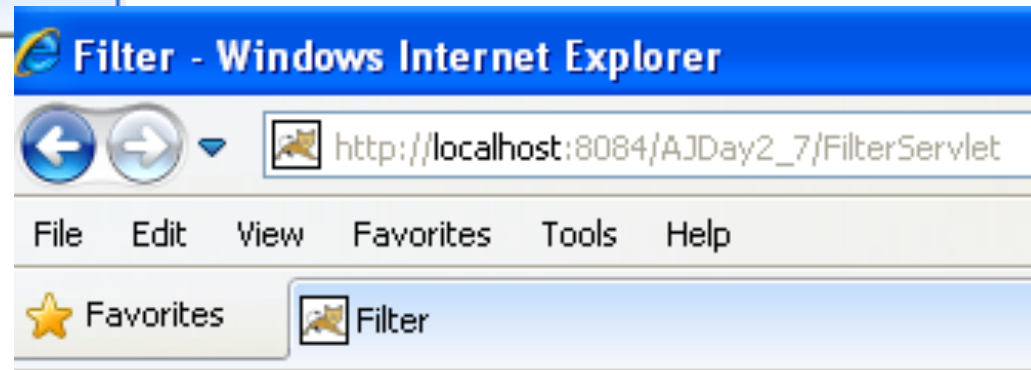
# Filter Example

- Building the web application shows as the following GUI in sequence



## Filter Demo

[Click here to see Filter Servlet](#)



## Filter Demo

KEY is First Filter

# Filter Example

filterDemo.html x

Preview

```

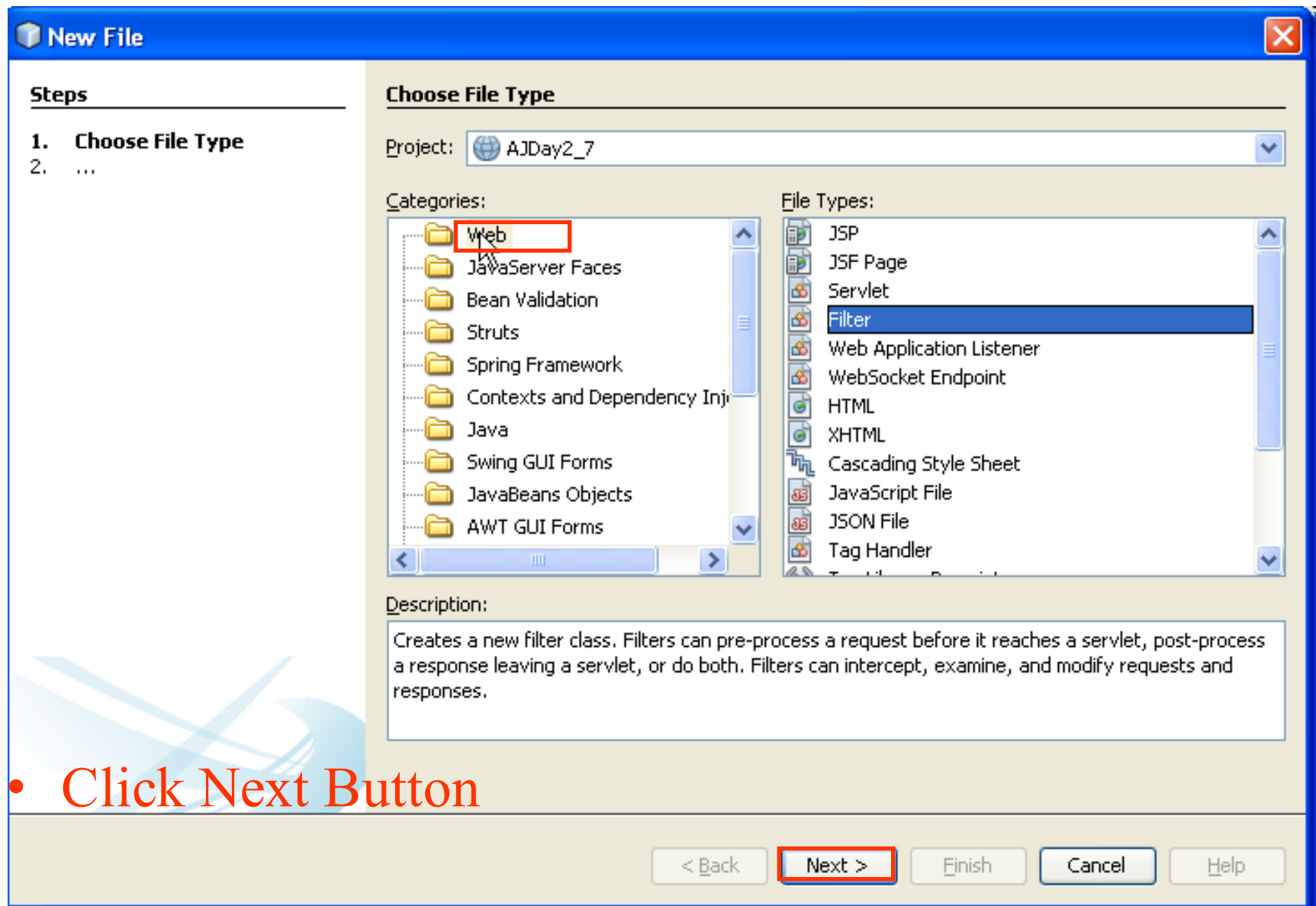
1  ...
5  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
6  <html>
7    <head>
8      <title>Filter</title>
9      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
10   </head>
11   <body>
12     <h1>Filter Demo</h1>
13
14     <a href="FilterServlet">Click here to see Filter Servlet</a>
15   </body>
16 </html>
  
```

# Filter Example

```

FilterServlet.java x
17  * @author Trong Khanh
18  */
19  public class FilterServlet extends HttpServlet {
20
21      /**...*/
22
23      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
24      throws ServletException, IOException {
25          response.setContentType("text/html;charset=UTF-8");
26          PrintWriter out = response.getWriter();
27          try {
28              out.println("<html>");
29              out.println("<head>");
30              out.println("<title>Filter</title>");
31              out.println("</head>");
32              out.println("<body>");
33              out.println("<h1>Filter Demo</h1>");
34
35              String test = (String) request.getAttribute("KEY");
36              out.println("KEY is " + test);
37
38              out.println("</body>");
39              out.println("</html>");
40          } finally {
41              out.close();
42          }
43      }
44  }
  
```

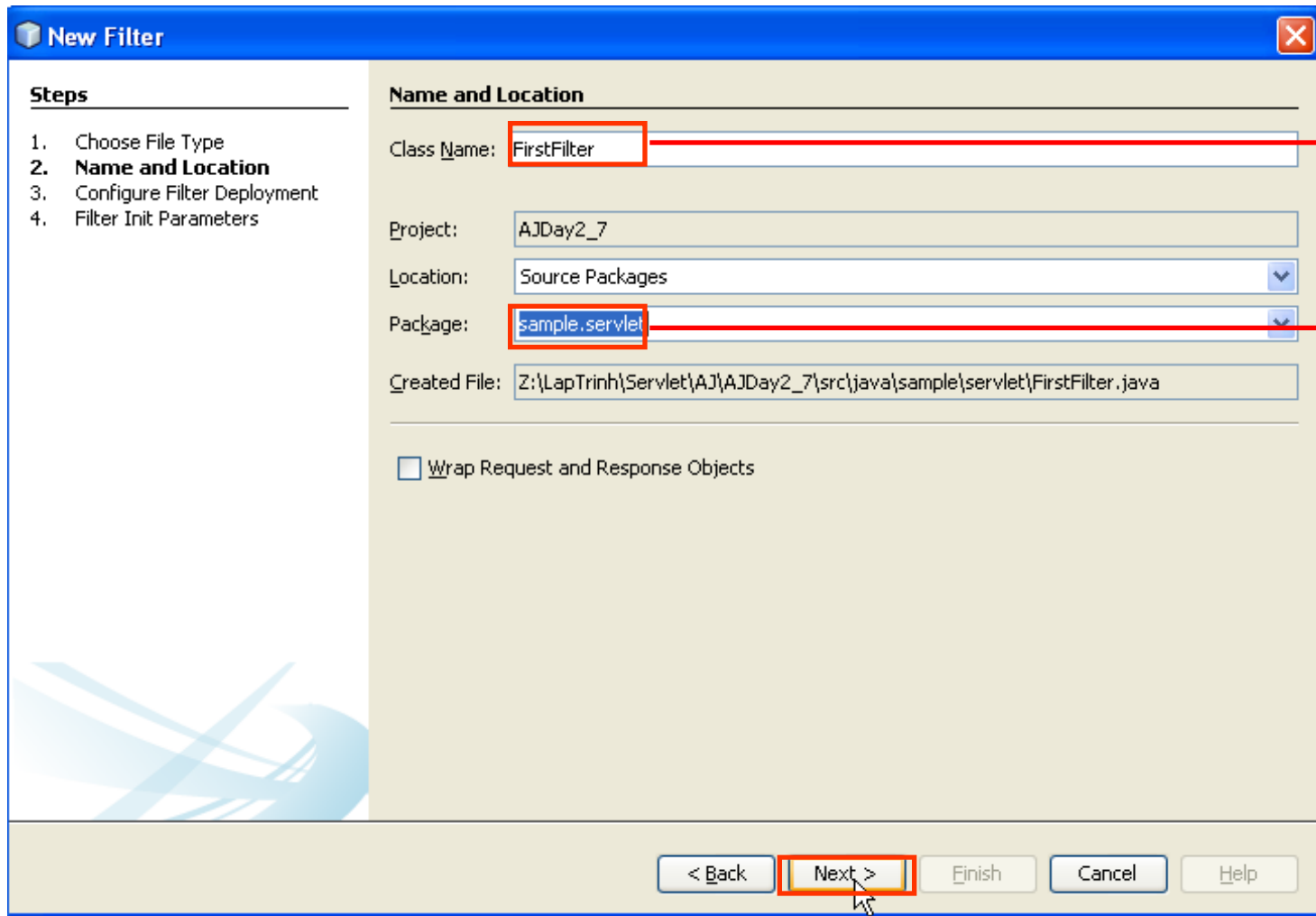
# Filter Example



- Click Next Button



# Filter Example



The image shows a 'New Filter' dialog box with a blue title bar and a close button. On the left, a 'Steps' list shows four steps: 1. Choose File Type, 2. Name and Location (highlighted), 3. Configure Filter Deployment, and 4. Filter Init Parameters. The main area is titled 'Name and Location' and contains several fields: 'Class Name' with the value 'FirstFilter' (highlighted with a red box and an arrow pointing to it with the text 'Fill your filter name'), 'Project' with the value 'AJDay2\_7', 'Location' with the value 'Source Packages' (a dropdown menu), and 'Package' with the value 'sample.servlet' (highlighted with a red box and an arrow pointing to it with the text 'Fill/choose package name'). Below these fields is a 'Created File' field showing the path 'Z:\LapTrinh\Servlet\AJ\AJDay2\_7\src\java\sample\ervlet\FirstFilter.java'. At the bottom, there is an unchecked checkbox labeled 'Wrap Request and Response Objects' and a row of buttons: '< Back', 'Next >' (highlighted with a red box and a mouse cursor), 'Finish', 'Cancel', and 'Help'.

Fill your filter name

Fill/choose package name

- Click Next Button

# Filter Example

**New Filter**

**Steps**

1. Choose File Type
2. Name and Location
3. **Configure Filter Deployment**
4. Filter Init Parameters

**Configure Filter Deployment**

Register the Filter with the application by giving the Filter an internal name. Describe when the Filter is invoked by listing the HTTP request path patterns or Servlets to which the Filter applies. Order this Filter's mappings relative to any other Filter invocation.

Class Name:

Filter Name:

Filter Mappings:

Filter name	Applies to
FirstFilter	/*

Buttons: New... Edit... Delete Move Up Move Down

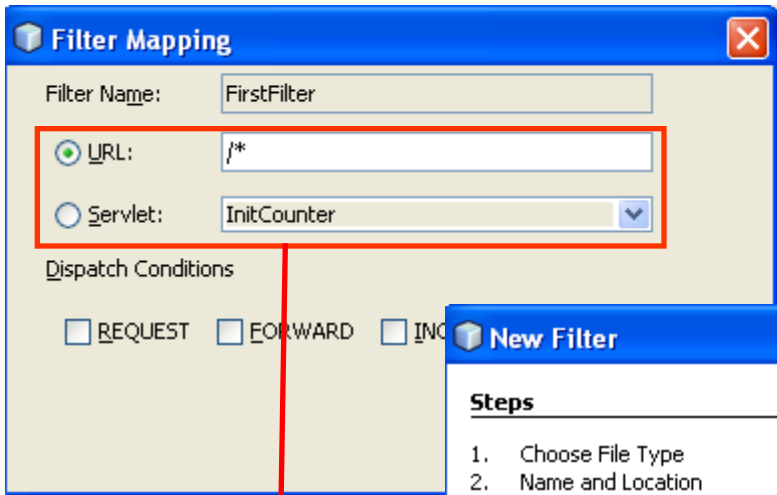
Navigation: < Back Next > Finish Cancel Help

Apply filter

Edit the Apply filter

- Click Edit Button to apply Filter the selected Servlet
- Otherwise, click Finish Button

# Filter Example



**Filter Mapping**

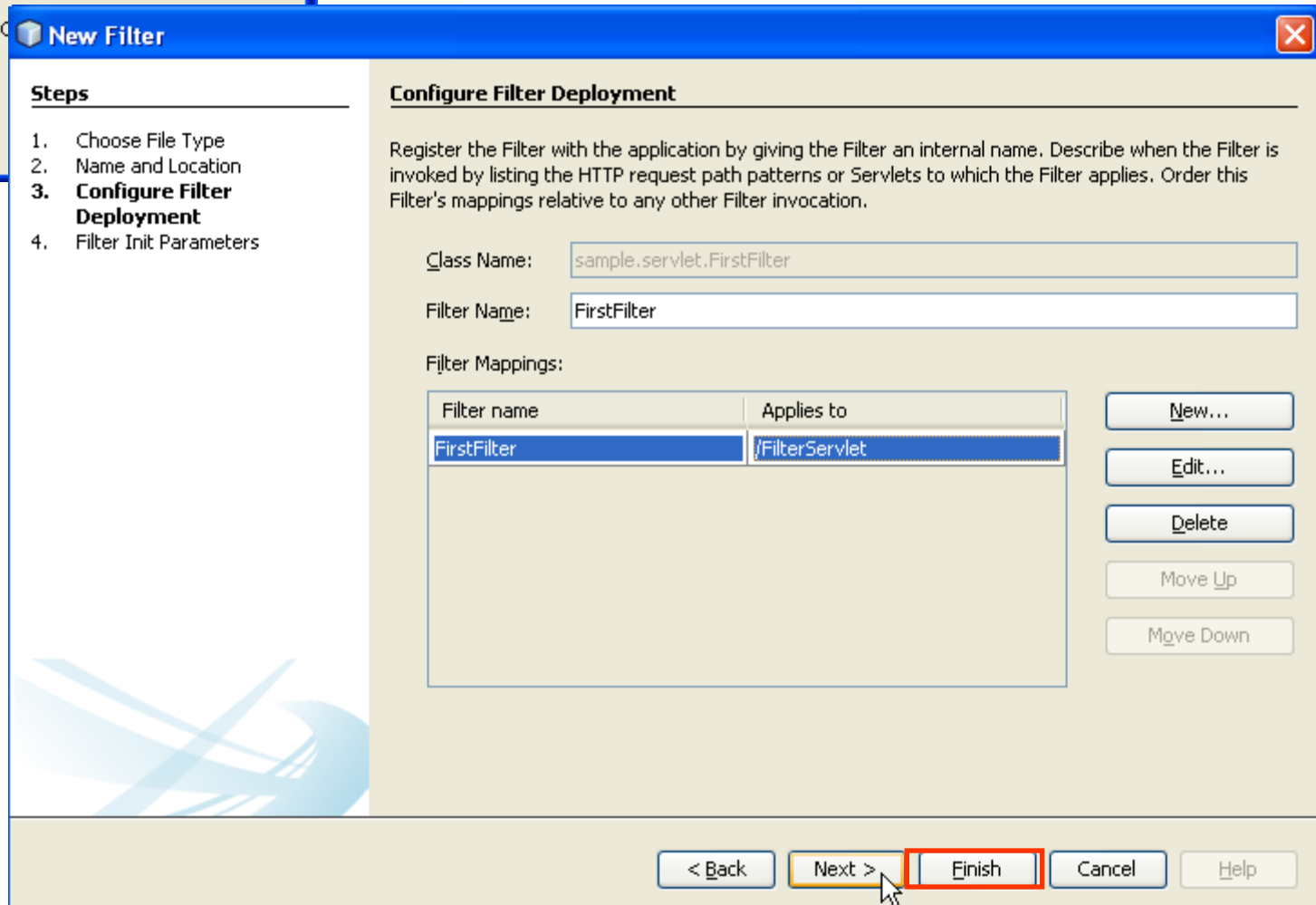
Filter Name:

☒ URL:

☐ Servlet:

Dispatch Conditions

☐ REQUEST ☐ FORWARD ☐ INC



**New Filter**

**Steps**

1. Choose File Type
2. Name and Location
3. **Configure Filter Deployment**
4. Filter Init Parameters

**Configure Filter Deployment**

Register the Filter with the application by giving the Filter an internal name. Describe when the Filter is invoked by listing the HTTP request path patterns or Servlets to which the Filter applies. Order this Filter's mappings relative to any other Filter invocation.

Class Name:

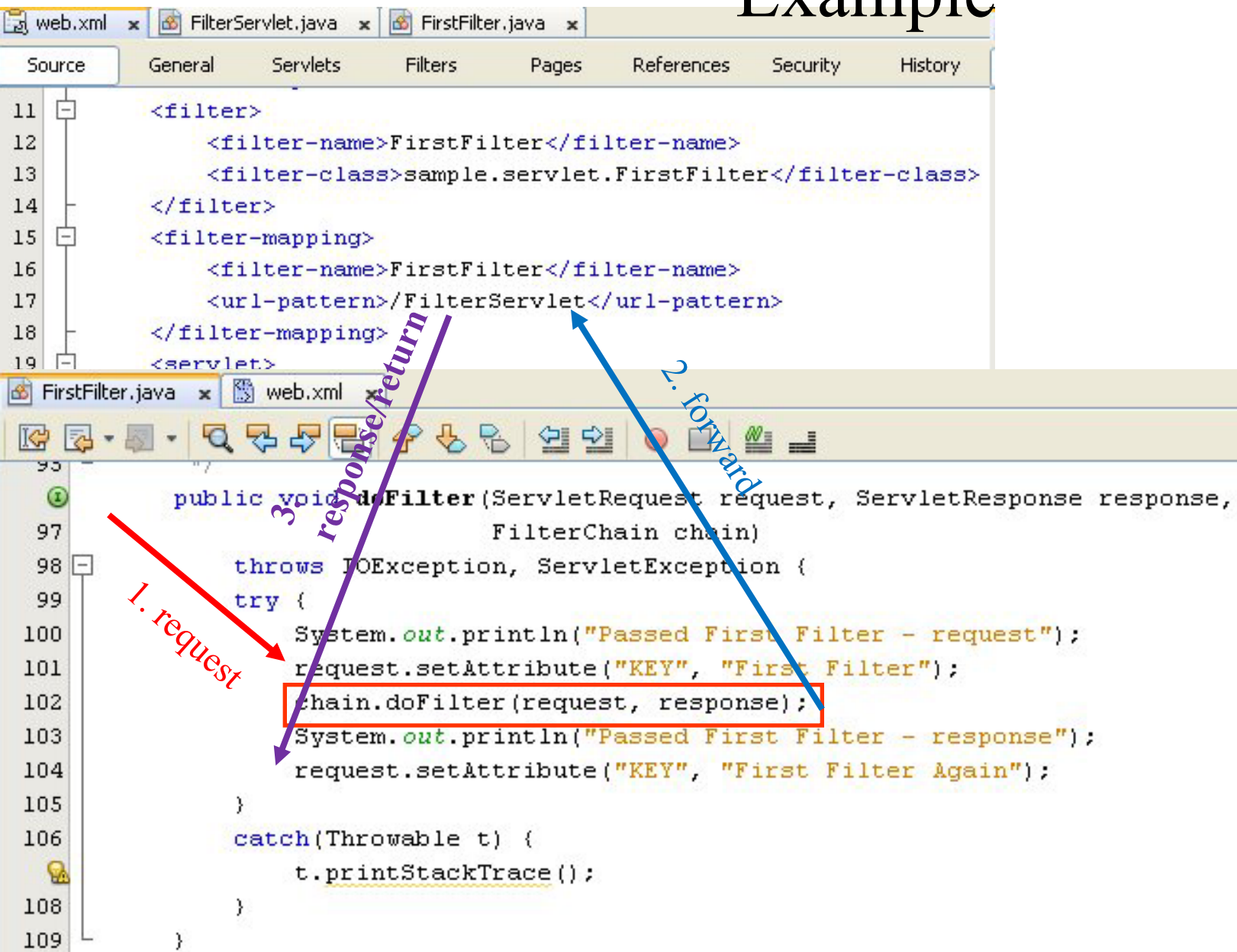
Filter Name:

Filter Mappings:

Filter name	Applies to
FirstFilter	/FilterServlet

Select the URL and typing the URL string, or Select the Servlet and choose the approximate Servlet in combo box

# Filter Example



The screenshot shows an IDE with two files open: `web.xml` and `FirstFilter.java`. The `web.xml` file defines a filter named `FirstFilter` that maps to the `FilterServlet` URL pattern. The `FirstFilter.java` file implements the `Filter` interface. The `doFilter` method is annotated with three numbered steps: 1. request (red arrow), 2. forward (blue arrow), and 3. response/return (purple arrow).

```

11 <filter>
12     <filter-name>FirstFilter</filter-name>
13     <filter-class>sample.servlet.FirstFilter</filter-class>
14 </filter>
15 <filter-mapping>
16     <filter-name>FirstFilter</filter-name>
17     <url-pattern>/FilterServlet</url-pattern>
18 </filter-mapping>
19 </servlet>

20 public void doFilter(ServletRequest request, ServletResponse response,
21                     FilterChain chain)
22     throws IOException, ServletException {
23     try {
24         System.out.println("Passed First Filter - request");
25         request.setAttribute("KEY", "First Filter");
26         chain.doFilter(request, response);
27         System.out.println("Passed First Filter - response");
28         request.setAttribute("KEY", "First Filter Again");
29     }
30     catch(Throwable t) {
31         t.printStackTrace();
32     }
33 }
  
```

Annotations in the image:

- 1. request** (red arrow) points to the `request` parameter in the `doFilter` method signature.
- 2. forward** (blue arrow) points to the `chain.doFilter(request, response);` line.
- 3. response/return** (purple arrow) points to the `response` parameter in the `doFilter` method signature.

# Filter Example

Output

Apache Tomcat 7.0.27.0 x Apache Tomcat 7.0.27.0 Log x AJDay2\_7 (run-deploy) x

```

thg 10 30, 2013 8:08:38 SA org.apache.catalina.core.ApplicationContext log
INFO: FirstFilter:Initializing filter

```

Output

Apache Tomcat 7.0.27.0 x Apache Tomcat 7.0.27.0 Log x AJDay2\_7 (run-deploy) x

```

Passed First Filter - request
Passed First Filter - response

```

web.xml x FilterServlet.java x FirstFilter.java x

Source

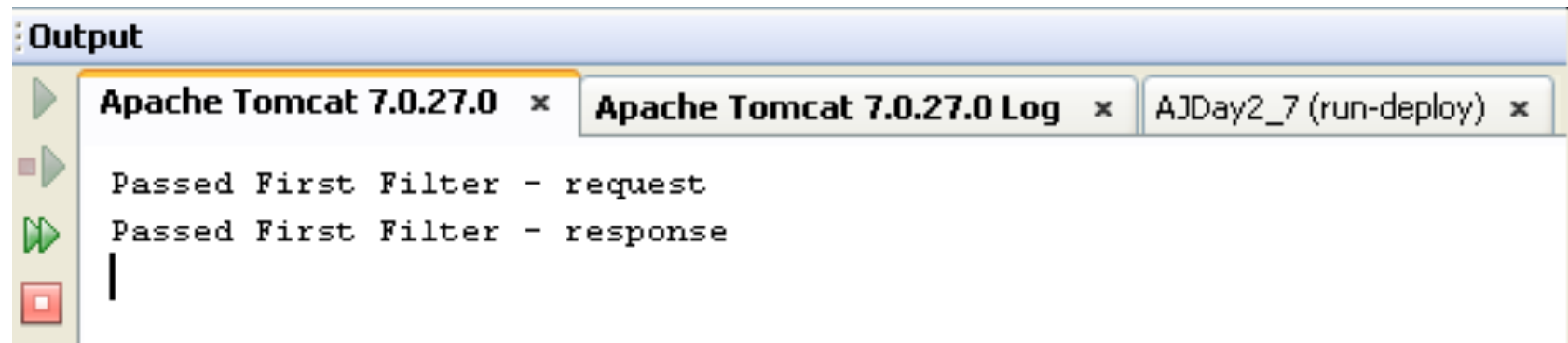
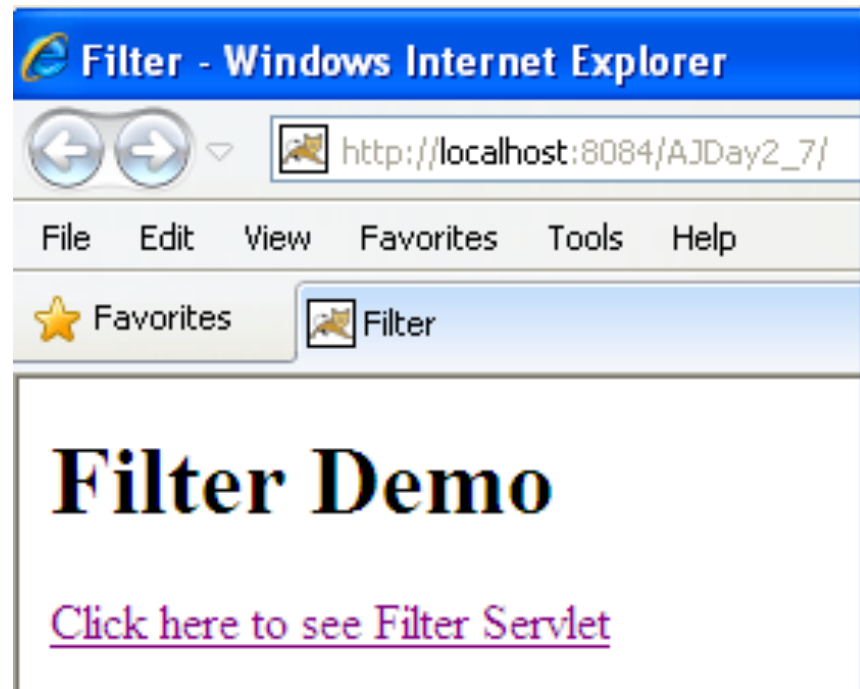
General Servlets Filters Pages References Security History

```

11 <filter>
12     <filter-name>FirstFilter</filter-name>
13     <filter-class>sample.servlet.FirstFilter</filter-class>
14 </filter>
15 <filter-mapping>
16     <filter-name>FirstFilter</filter-name>
17     <url-pattern>/*</url-pattern>
18 </filter-mapping>

```

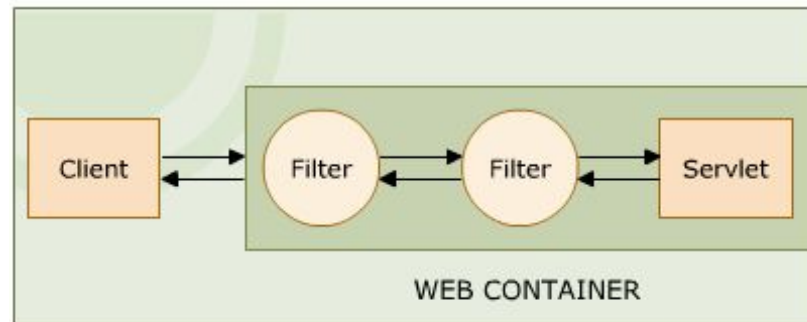
# Filter Example



# Filter Chain

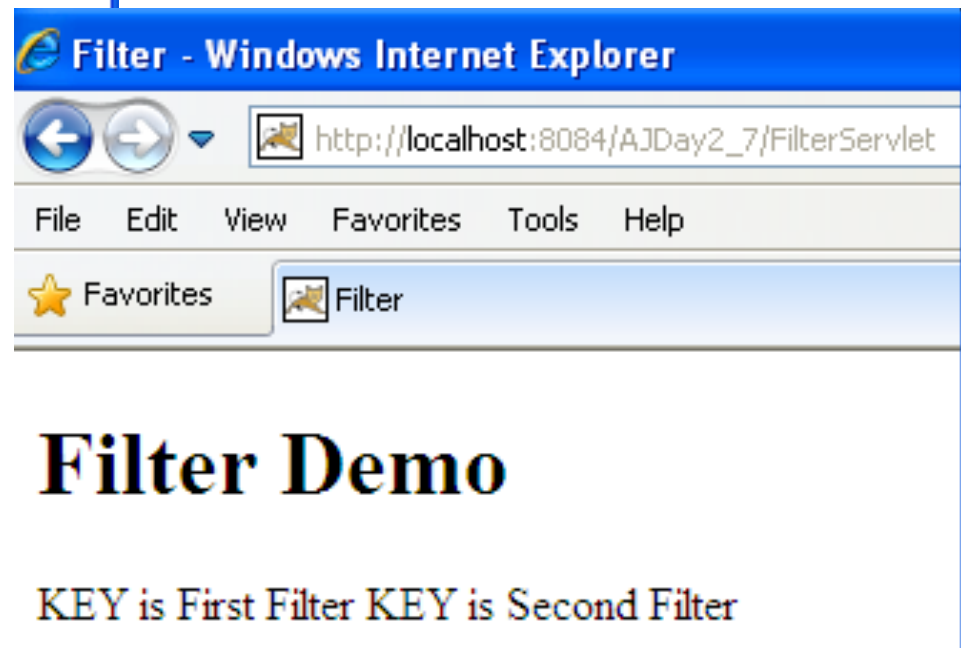
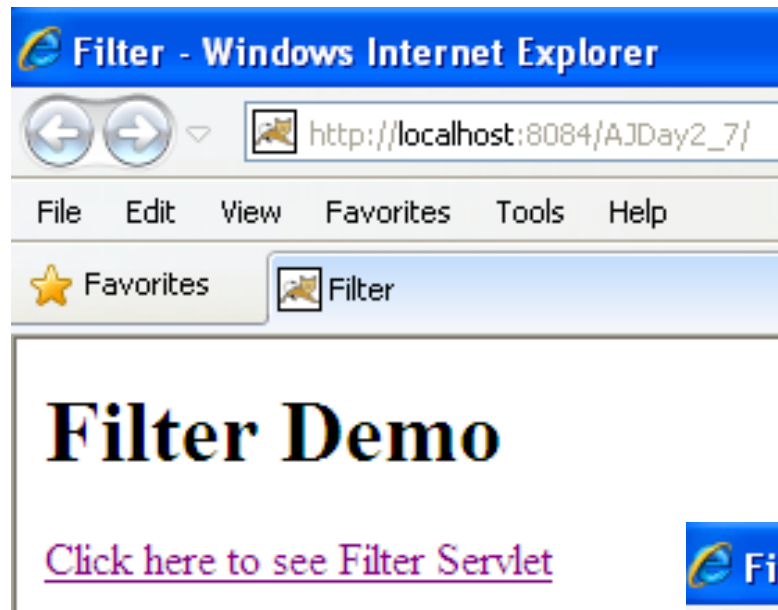
## Definition

- There can be **more than one filter** between the user and the endpoint – Invoke a **series of filters**
- A request or a response is **passed through one** filter to the **next** in the filter chain. So each request and response has to be serviced by each filter forming a filter chain
- If the Calling filter is last filter, will invoke web resource
- **FilterChain Interface**
  - Provides an object through the web container
  - The object invokes the next filter in a filter chain starting from the first filter from a particular end. If the calling filter is the last filter in the chain, it will invoke the web resource, such as JSP and servlet.
  - Only implement doFilter() method.
  - Forces the next filter in the chain to be invoked



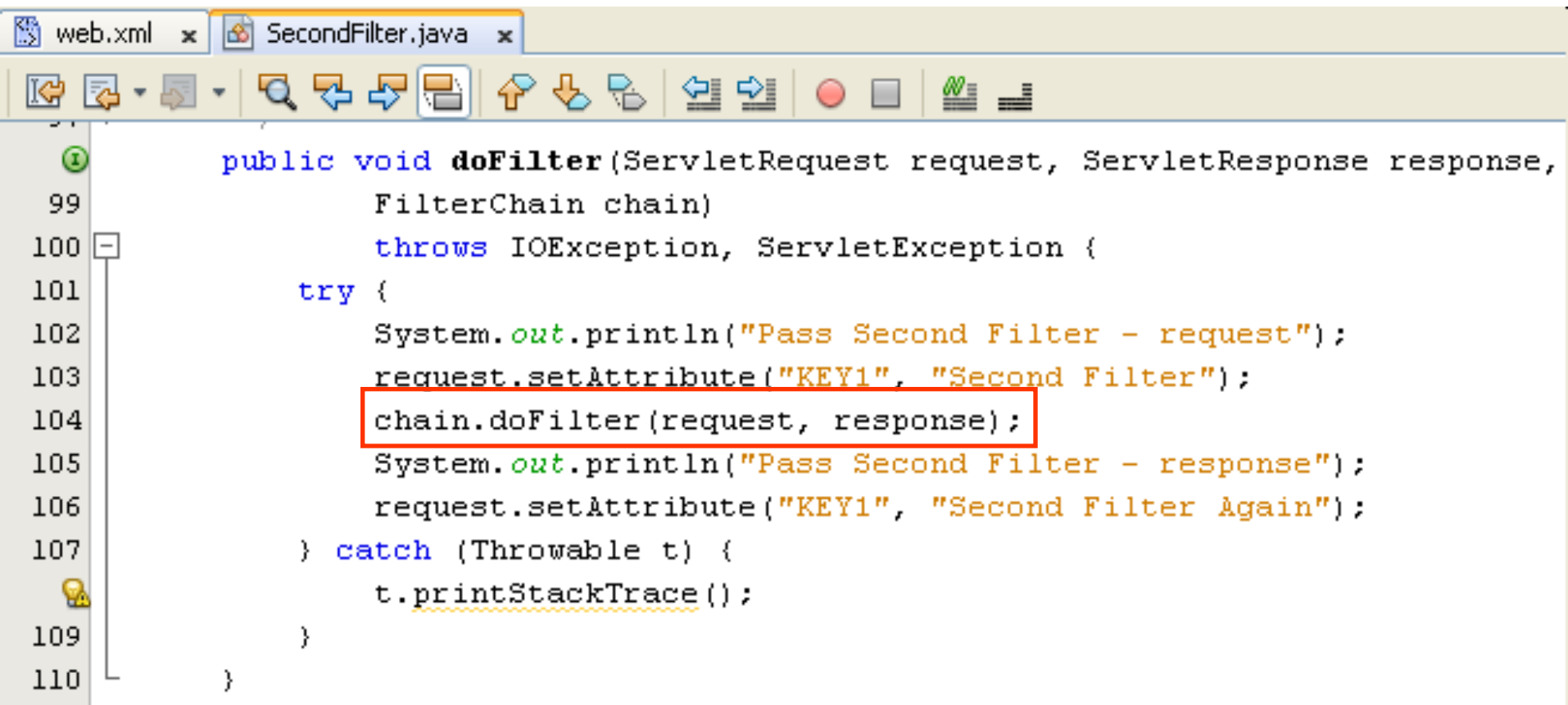
**Filter Chain**

# Filter Chain Example





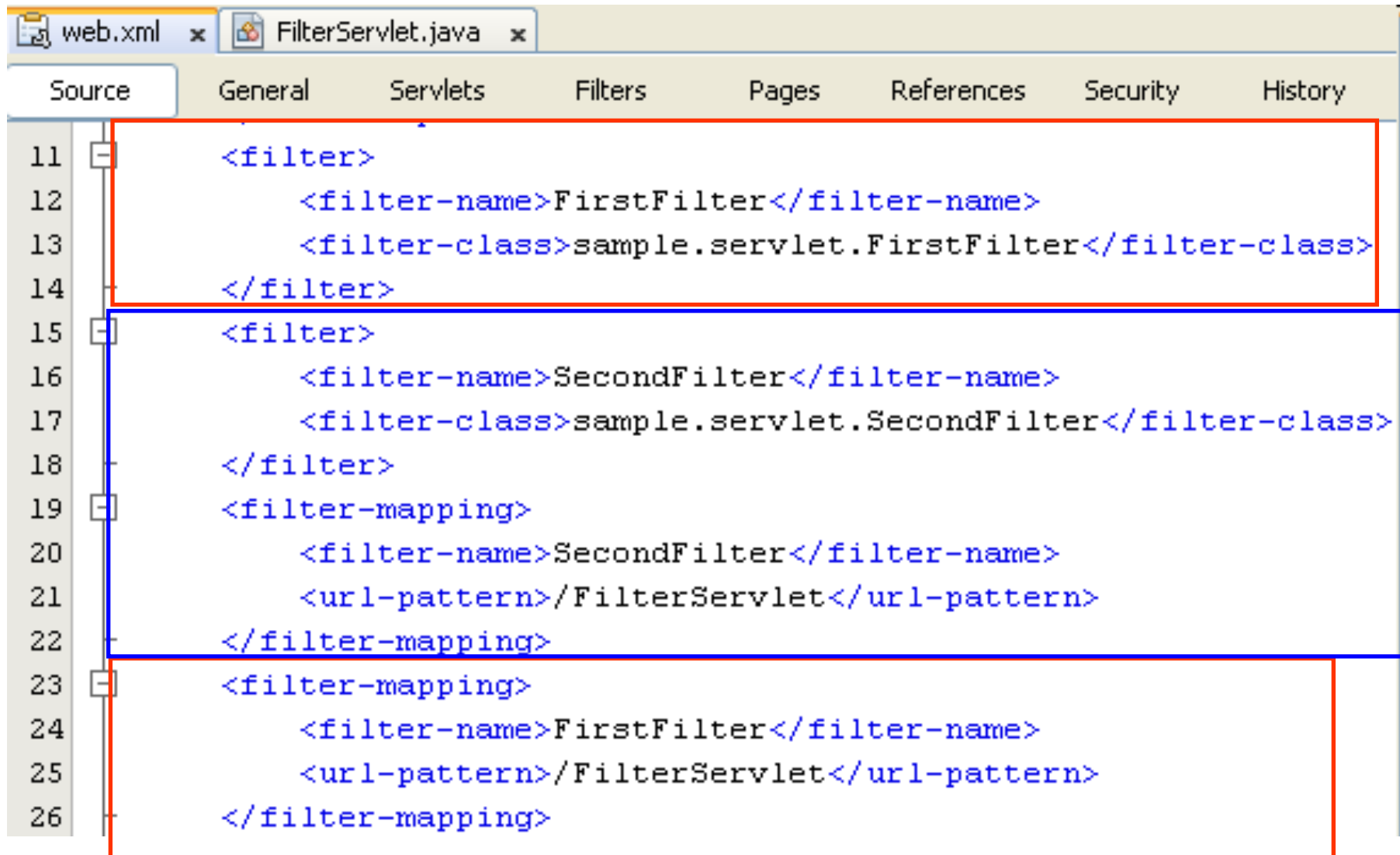
# Filter Chain Example



```

web.xml x SecondFilter.java x
public void doFilter(ServletRequest request, ServletResponse response,
    FilterChain chain)
    throws IOException, ServletException {
    try {
        System.out.println("Pass Second Filter - request");
        request.setAttribute("KEY1", "Second Filter");
        chain.doFilter(request, response);
        System.out.println("Pass Second Filter - response");
        request.setAttribute("KEY1", "Second Filter Again");
    } catch (Throwable t) {
        t.printStackTrace();
    }
}
  
```

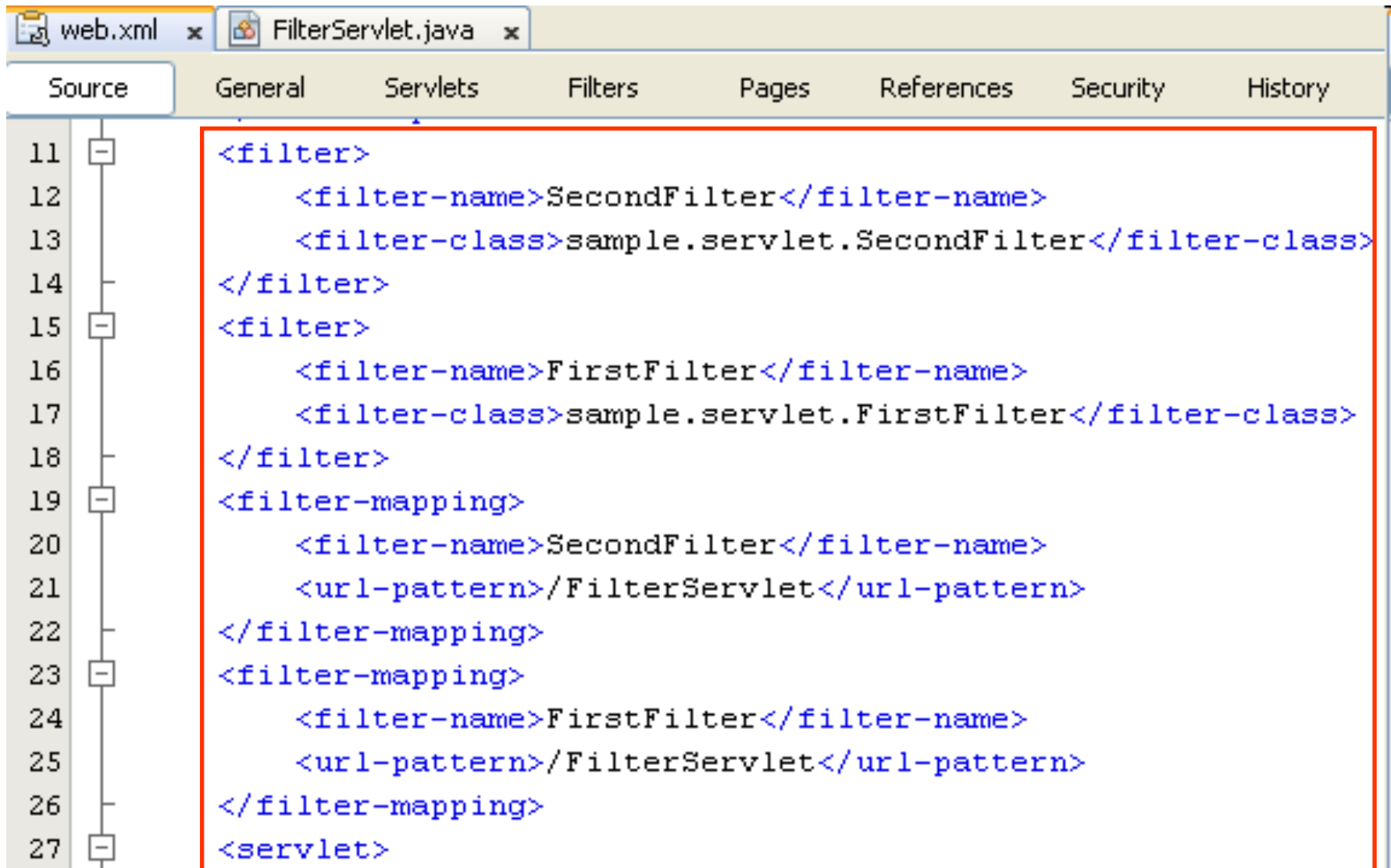
# Filter Chain Example



```

11 <filter>
12     <filter-name>FirstFilter</filter-name>
13     <filter-class>sample.servlet.FirstFilter</filter-class>
14 </filter>
15 <filter>
16     <filter-name>SecondFilter</filter-name>
17     <filter-class>sample.servlet.SecondFilter</filter-class>
18 </filter>
19 <filter-mapping>
20     <filter-name>SecondFilter</filter-name>
21     <url-pattern>/FilterServlet</url-pattern>
22 </filter-mapping>
23 <filter-mapping>
24     <filter-name>FirstFilter</filter-name>
25     <url-pattern>/FilterServlet</url-pattern>
26 </filter-mapping>
  
```

# Filter Chain Example

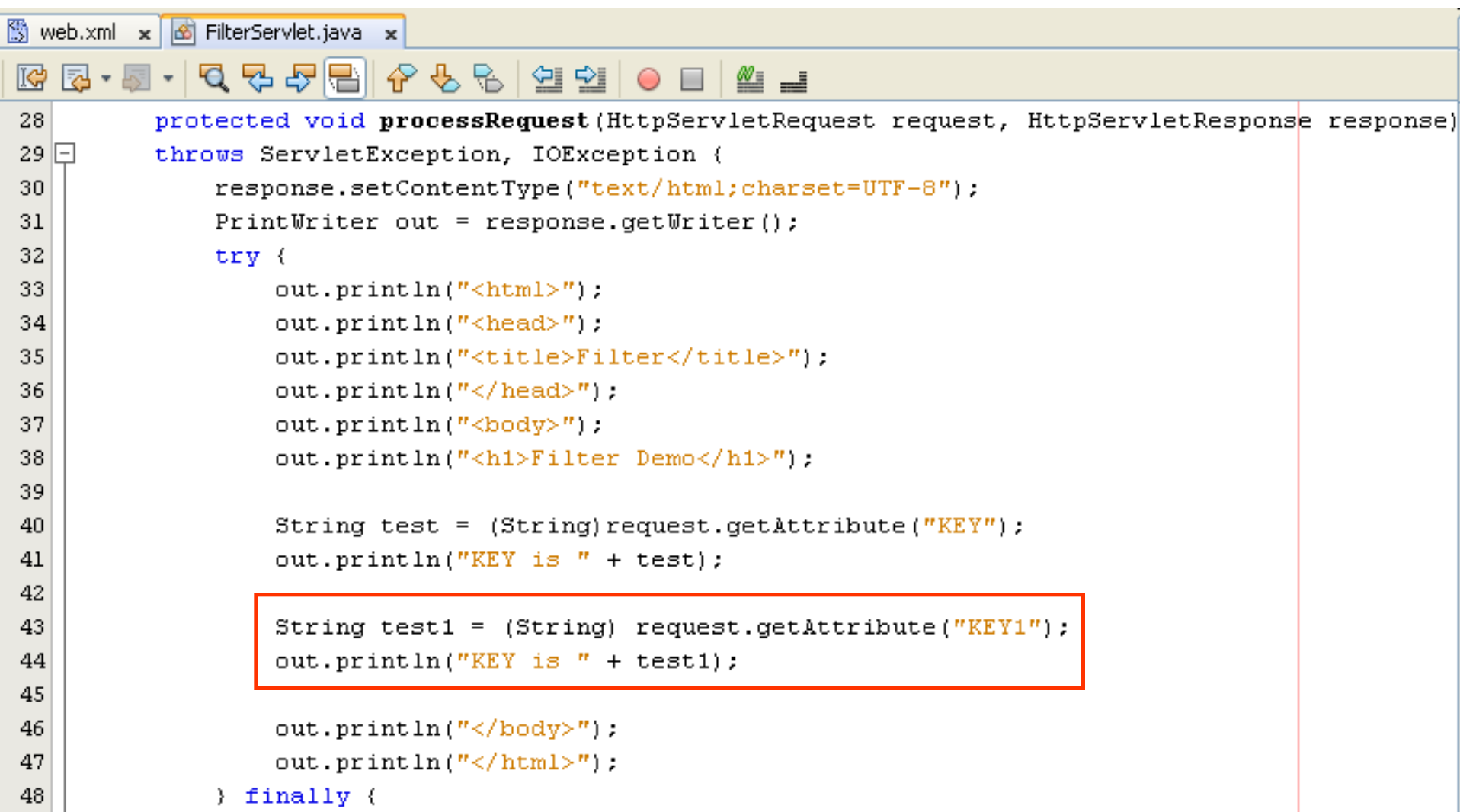


The screenshot shows an IDE window with two tabs: 'web.xml' and 'FilterServlet.java'. The 'web.xml' tab is active, and the 'Source' editor is displaying the XML configuration. The configuration defines two filters, 'SecondFilter' and 'FirstFilter', and two filter-mappings for the 'FilterServlet'. The filter-mappings are ordered such that 'SecondFilter' is applied first, followed by 'FirstFilter', creating a filter chain. The entire XML content is enclosed in a red rectangular box.

```

11 <filter>
12     <filter-name>SecondFilter</filter-name>
13     <filter-class>sample.servlet.SecondFilter</filter-class>
14 </filter>
15 <filter>
16     <filter-name>FirstFilter</filter-name>
17     <filter-class>sample.servlet.FirstFilter</filter-class>
18 </filter>
19 <filter-mapping>
20     <filter-name>SecondFilter</filter-name>
21     <url-pattern>/FilterServlet</url-pattern>
22 </filter-mapping>
23 <filter-mapping>
24     <filter-name>FirstFilter</filter-name>
25     <url-pattern>/FilterServlet</url-pattern>
26 </filter-mapping>
27 <servlet>
  
```

# Filter Chain Example



```

28  protected void processRequest(HttpServletRequest request, HttpServletResponse response)
29  throws ServletException, IOException {
30      response.setContentType("text/html;charset=UTF-8");
31      PrintWriter out = response.getWriter();
32      try {
33          out.println("<html>");
34          out.println("<head>");
35          out.println("<title>Filter</title>");
36          out.println("</head>");
37          out.println("<body>");
38          out.println("<h1>Filter Demo</h1>");
39
40          String test = (String)request.getAttribute("KEY");
41          out.println("KEY is " + test);
42
43          String test1 = (String) request.getAttribute("KEY1");
44          out.println("KEY is " + test1);
45
46          out.println("</body>");
47          out.println("</html>");
48      } finally {
  
```

# Filter Chain Example

**Output**

Apache Tomcat 7.0.27.0 x Apache Tomcat 7.0.27.0 Log x AJDay2\_7 (run-deploy) x

```
thg 10 30, 2013 8:11:16 SA org.apache.catalina.core.ApplicationContext log
INFO: FirstFilter:Initializing filter
thg 10 30, 2013 8:11:16 SA org.apache.catalina.core.ApplicationContext log
INFO: SecondFilter:Initializing filter
```

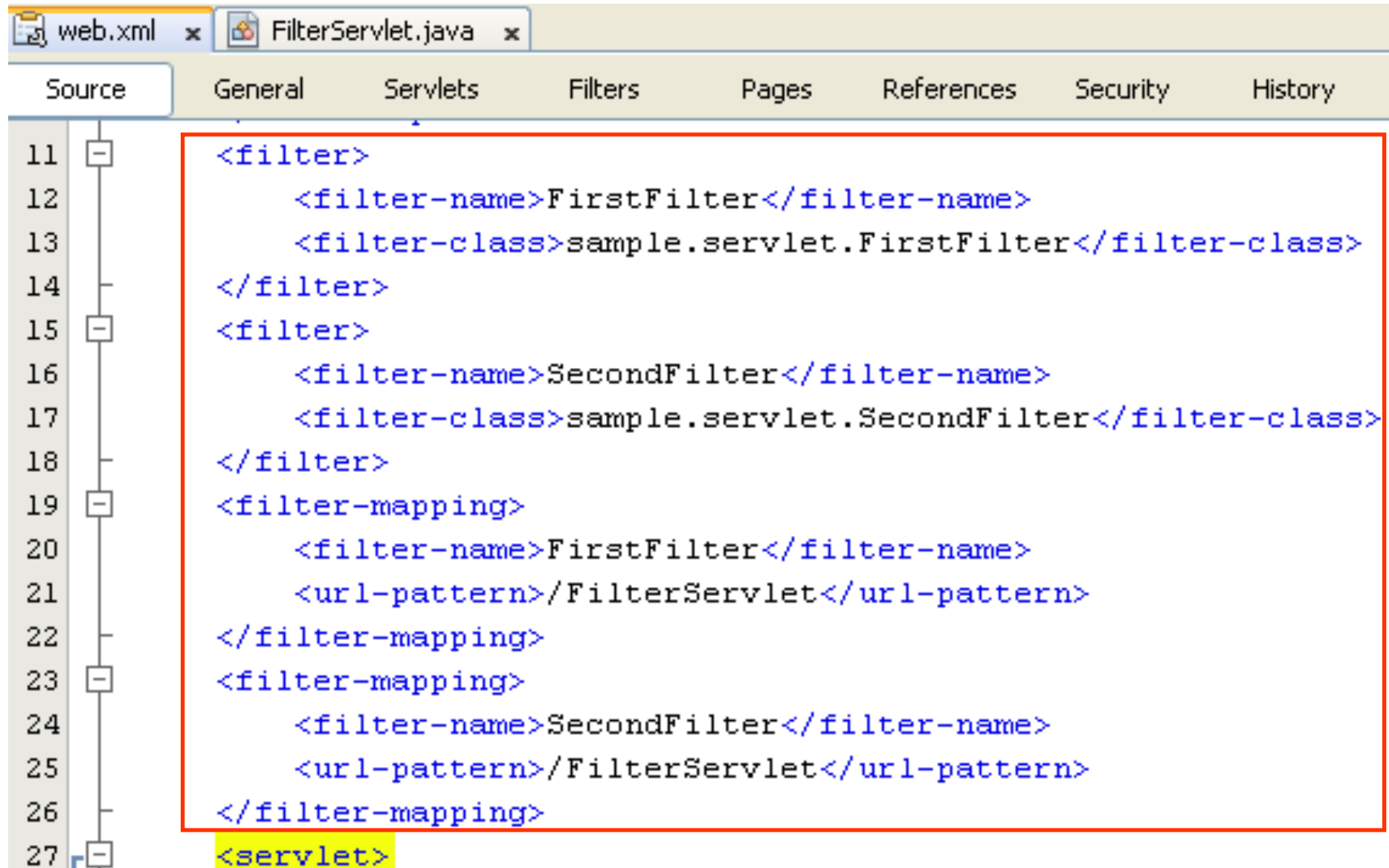
**Output**

Apache Tomcat 7.0.27.0 x Apache Tomcat 7.0.27.0 Log x AJDay2\_7 (run-deploy) x

```
Pass Second Filter - request
Passed First Filter - request
Passed First Filter - response
Pass Second Filter - response
```

# Filter Chain

## Example – Change position



The screenshot shows an IDE with two tabs: 'web.xml' and 'FilterServlet.java'. The 'web.xml' tab is active, and the 'Filters' perspective is selected. The XML content is as follows:

```

11 <filter>
12     <filter-name>FirstFilter</filter-name>
13     <filter-class>sample.servlet.FirstFilter</filter-class>
14 </filter>
15 <filter>
16     <filter-name>SecondFilter</filter-name>
17     <filter-class>sample.servlet.SecondFilter</filter-class>
18 </filter>
19 <filter-mapping>
20     <filter-name>FirstFilter</filter-name>
21     <url-pattern>/FilterServlet</url-pattern>
22 </filter-mapping>
23 <filter-mapping>
24     <filter-name>SecondFilter</filter-name>
25     <url-pattern>/FilterServlet</url-pattern>
26 </filter-mapping>
27 <servlet>

```

A red rectangular box highlights the filter mappings from line 19 to line 26. In this configuration, the FirstFilter is mapped to /FilterServlet before the SecondFilter, which is the opposite of the order defined in the filter declarations above.

# Filter Chain Example

**Output**

Apache Tomcat 7.0.27.0 x Apache Tomcat 7.0.27.0 Log x AJDay2\_7 (run-deploy) x

```
thg 10 30, 2013 8:06:32 SA org.apache.catalina.core.ApplicationContext log
INFO: FirstFilter:Initializing filter
thg 10 30, 2013 8:06:32 SA org.apache.catalina.core.ApplicationContext log
INFO: SecondFilter:Initializing filter
```

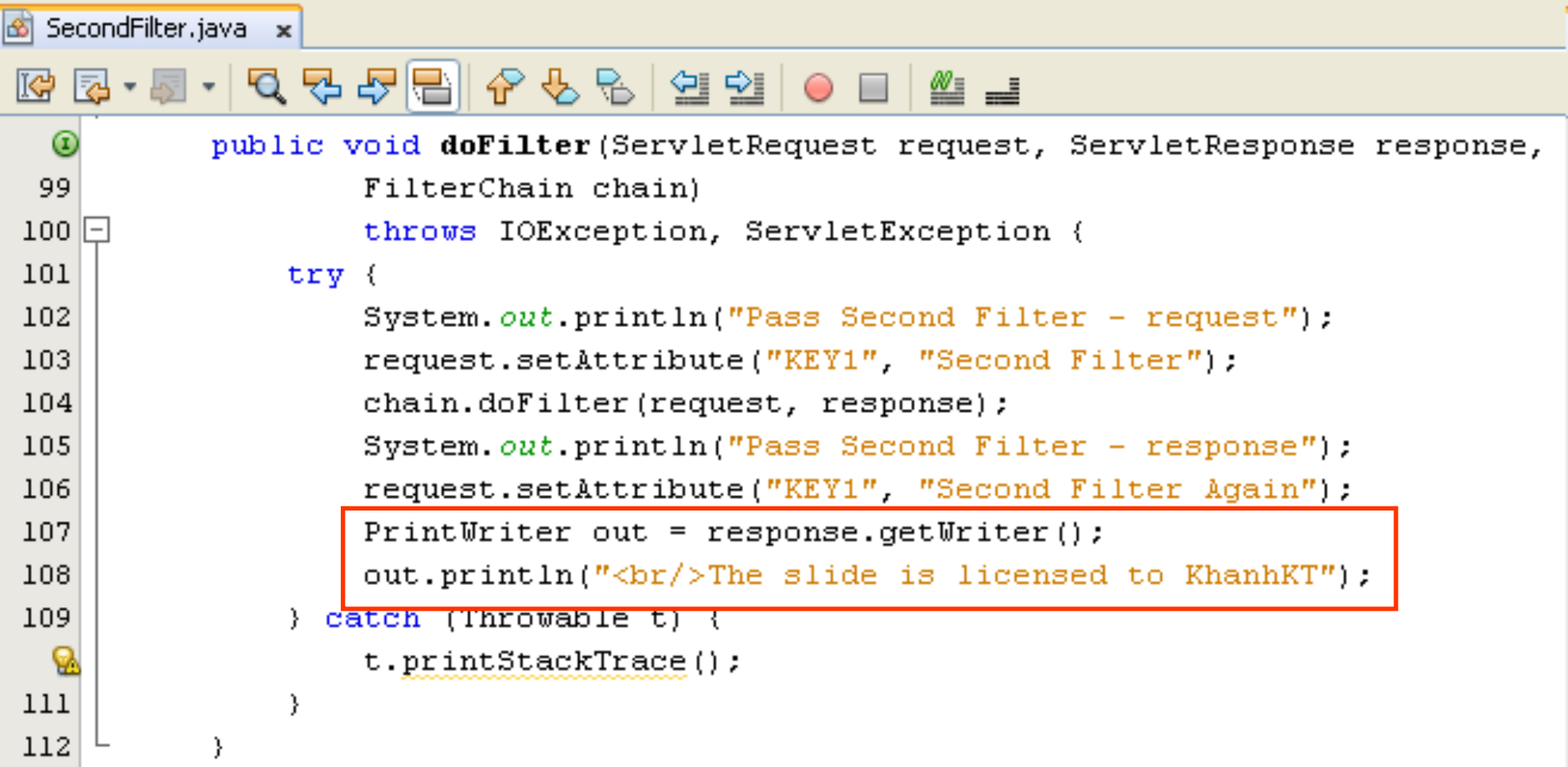
**Output**

▶ Apache Tomcat 7.0.27.0 x ▶ Apache Tomcat 7.0.27.0 Log x ▶ AJDay2\_7 (run-deploy) x

```
Passed First Filter - request
Pass Second Filter - request
Pass Second Filter - response
Passed First Filter - response
```

# Filter Chain

## Why need a Wrapper Class



```

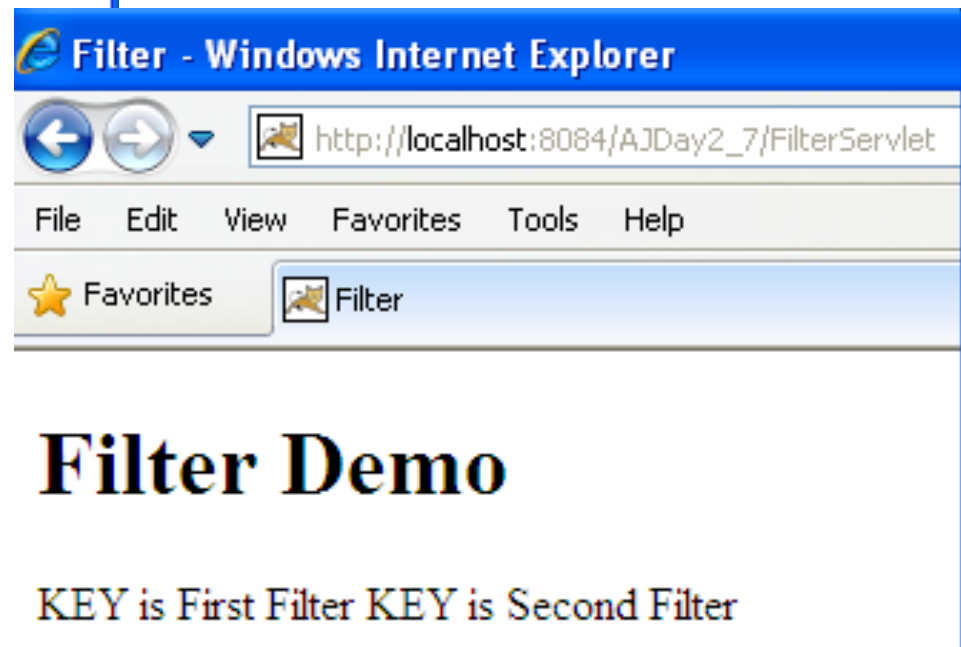
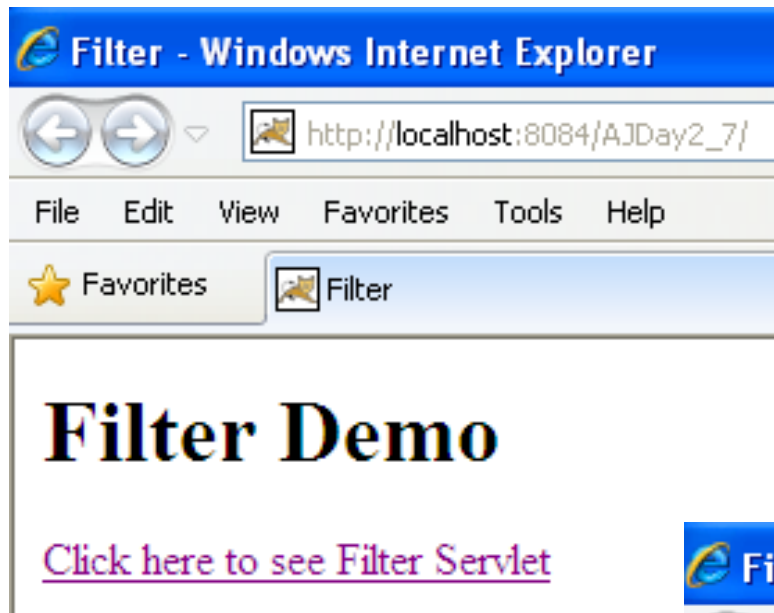
99     public void doFilter(ServletRequest request, ServletResponse response,
100                          FilterChain chain)
101                          throws IOException, ServletException {
102     try {
103         System.out.println("Pass Second Filter - request");
104         request.setAttribute("KEY1", "Second Filter");
105         chain.doFilter(request, response);
106         System.out.println("Pass Second Filter - response");
107         request.setAttribute("KEY1", "Second Filter Again");
108         PrintWriter out = response.getWriter();
109         out.println("<br/>The slide is licensed to KhanhKT");
110     } catch (Throwable t) {
111         t.printStackTrace();
112     }

```



# Filter Chain

## Why need a Wrapper Class

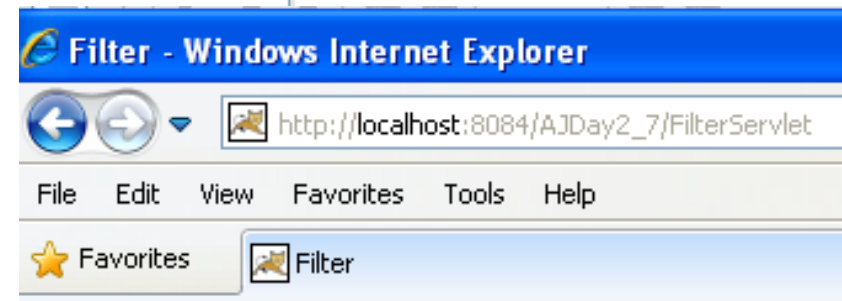


# Filter Chain

## Why need a Wrapper Class

```

28  protected void processRequest(HttpServletRequest request, HttpServletResponse response)
29  throws ServletException, IOException {
30      response.setContentType("text/html;charset=UTF-8");
31      PrintWriter out = response.getWriter();
32      try {
33          out.println("<html>");
34          out.println("<head>");
35          out.println("<title>Filter</title>");
36          out.println("</head>");
37          out.println("<body>");
38          out.println("<h1>Filter Demo</h1>");
39
40          String test = (String)request.getAttribute("key");
41          out.println("KEY is " + test);
42
43          String test1 = (String) request.getAttribute("key");
44          out.println("KEY is " + test1);
45
46          out.println("</body>");
47          out.println("</html>");
48      } finally {
49          //out.close();
50      }
51  }
    
```



## Filter Demo

KEY is First Filter KEY is Second Filter

The slide is licensed to KhanhKT

# Filter Chain Wrapper Class

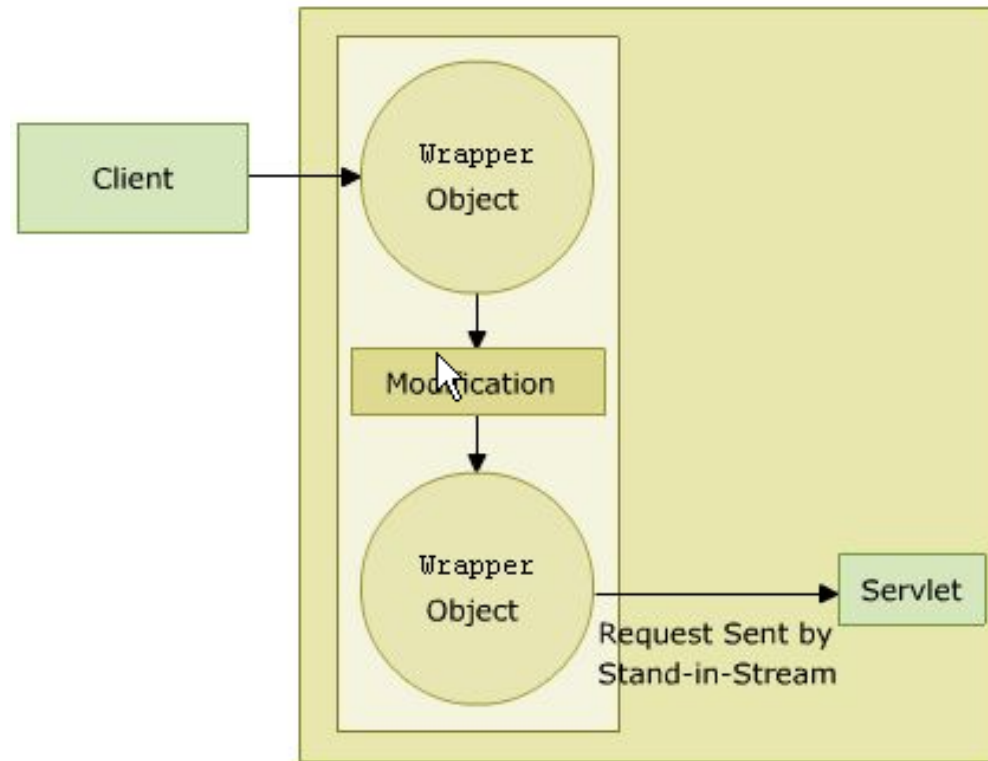
- To **modify or intercept** the request or response **before they can reach their logical destination**, the **required object can dynamically capture** the request or response
- Wrapper class
  - Creates the object to **capture the request and response** before they reach server and client respectively
  - The wrapper object generated by the filter **implements the `getWriter()` and `getOutputStream()`**, which returns a stand-in-stream. The stand-in-stream is passed to the servlet through the wrapper object
  - The wrapper object captures **the response through the stand-in-stream** and sends it back to the filter

Classes	Descriptions
<b>ServletRequestWrapper</b>	<ul style="list-style-type: none"><li>- Provides a convenient implementation of the ServletRequest interface</li><li>- Can be sub-classed by developers wishing to send the request to a servlet</li><li>- To override request methods, one should wrap the request in an object that extends ServletRequestWrapper or HttpServletRequestWrapper</li></ul>
<b>ServletResponseWrapper</b>	<ul style="list-style-type: none"><li>- Provides a convenient implementation of the ServletResponse interface</li><li>- Can be sub classed by developers wishing to send the response from a servlet.</li></ul>

# Filter Chain

## Wrapper Class – Altering Request

- Create filter class extends to the **ServletRequestWrapper** or **HttpServletRequestWrapper** class.
- The object captures the **HttpRequest** object from the client and sends it to the filters
- Through the objects filter extends some services to the request.

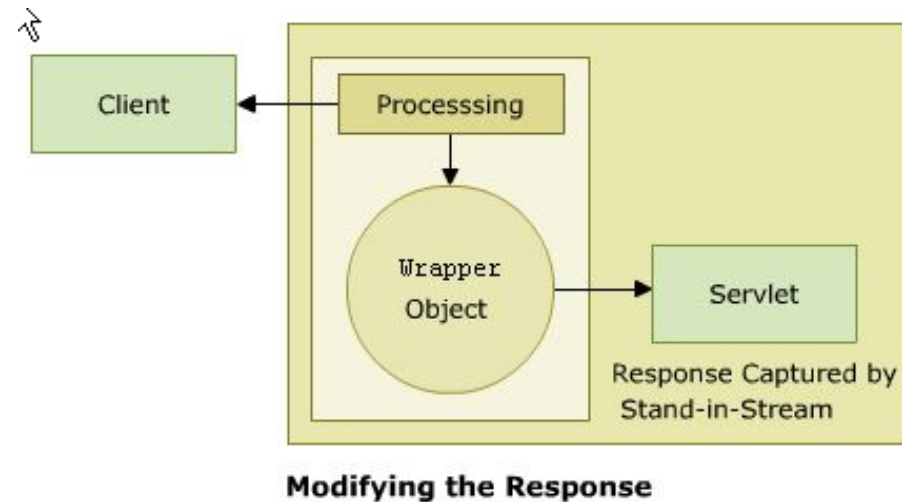


**Modifying the Request**

# Filter Chain

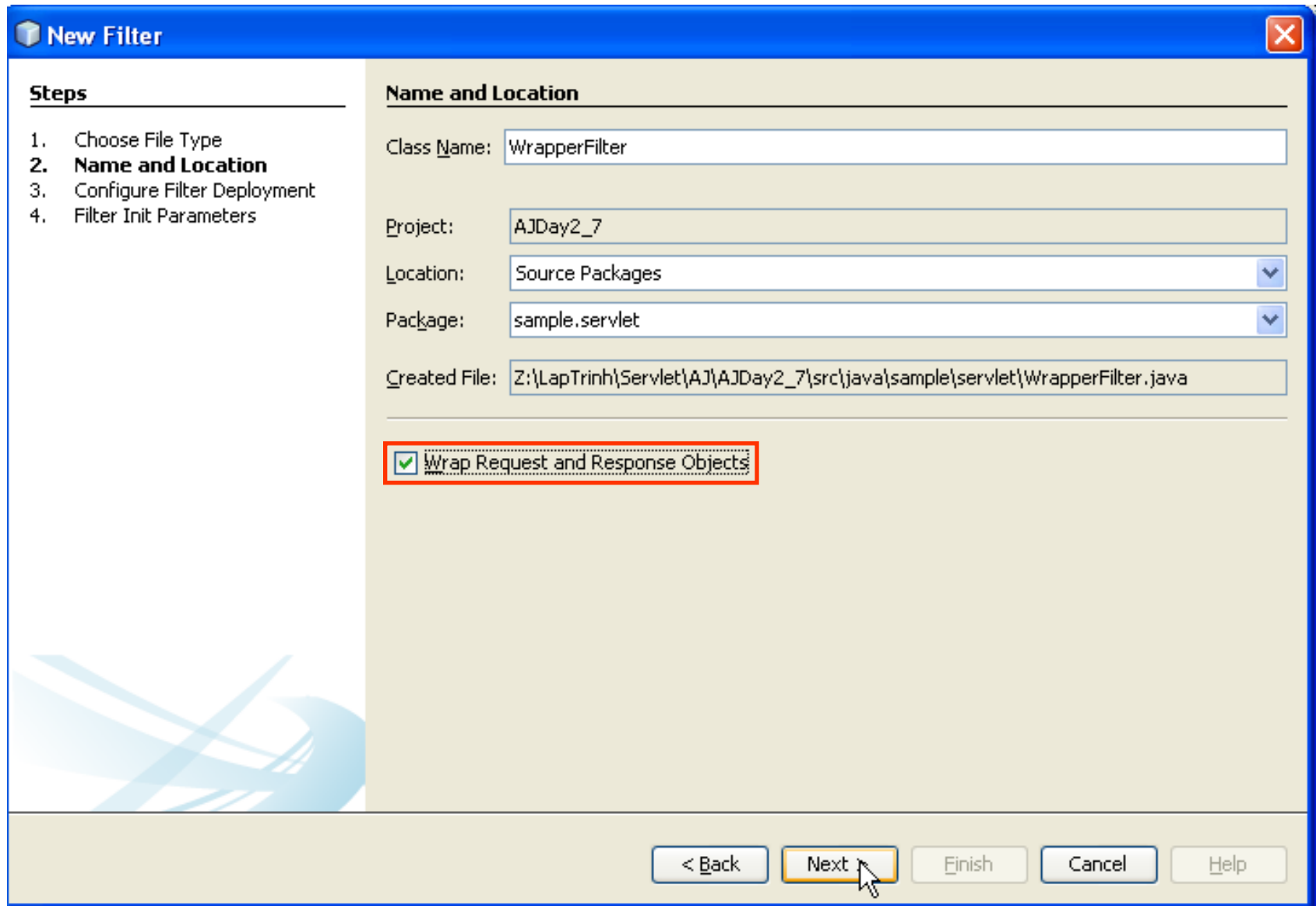
## Wrapper Class – Altering Response

- Create filter class extends to the **ServletResponseWrapper** or **HttpServletResponseWrapper** class.
- The object captures the **HttpServletRequest** object from the client and sends it to the filters
- Through the objects filter extends some services to the request.



# Filter Chain

## Wrapper Class – Example



The image shows a 'New Filter' wizard dialog box with a blue title bar and a close button. On the left, a 'Steps' pane lists four steps: 1. Choose File Type, 2. Name and Location (highlighted), 3. Configure Filter Deployment, and 4. Filter Init Parameters. The main area is titled 'Name and Location' and contains several input fields: 'Class Name' with the text 'WrapperFilter', 'Project' with 'AJDay2\_7', 'Location' with a dropdown menu showing 'Source Packages', and 'Package' with a dropdown menu showing 'sample.servlet'. Below these is a text field for 'Created File' containing the path 'Z:\LapTrinh\Servlet\AJ\AJDay2\_7\src\java\sample\servlet\WrapperFilter.java'. At the bottom of the main area, there is a checkbox labeled 'Wrap Request and Response Objects' which is checked and highlighted with a red rectangle. The bottom of the dialog features a row of buttons: '< Back', 'Next >' (highlighted with a mouse cursor), 'Finish', 'Cancel', and 'Help'.

**New Filter**

**Steps**

1. Choose File Type
- 2. Name and Location**
3. Configure Filter Deployment
4. Filter Init Parameters

**Name and Location**

Class Name:

Project:

Location:

Package:



Created File:

☒ Wrap Request and Response Objects

< Back   Next >   Finish   Cancel   Help

# Filter Chain

## Wrapper Class – Example


**New Filter**


**Steps**

1. Choose File Type
2. Name and Location
- 3. Configure Filter Deployment**
4. Filter Init Parameters

**Configure Filter Deployment**

Register the Filter with the application by giving the Filter an internal name. Describe when the Filter is invoked by listing the HTTP request path patterns or Servlets to which the Filter applies. Order this Filter's mappings relative to any other Filter invocation.

Class Name:

Filter Name:

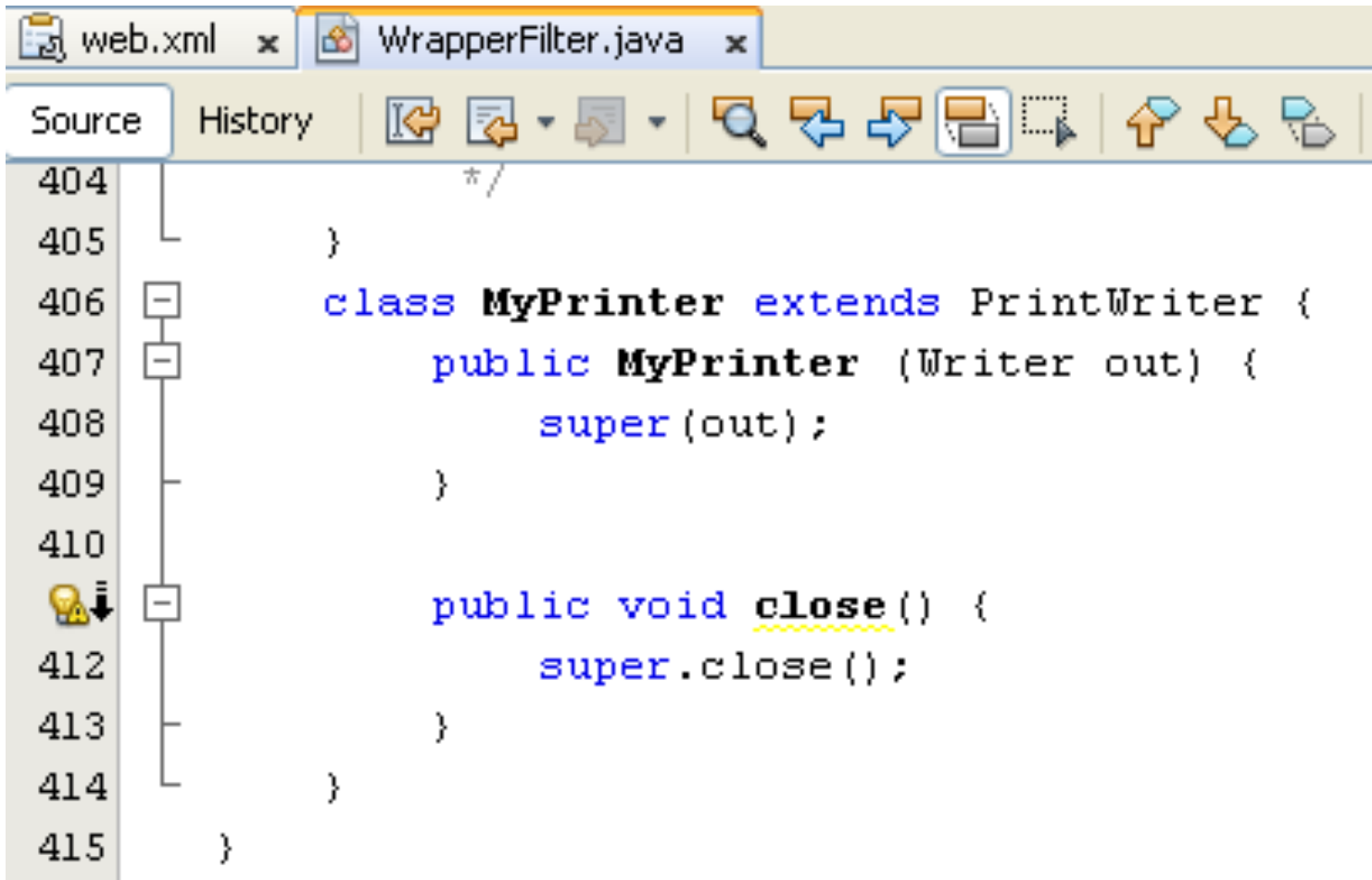
Filter Mappings:

Filter name	Applies to
WrapperFilter	/FilterServlet
FirstFilter	/FilterServlet
SecondFilter	/FilterServlet

# Filter Chain

## Wrapper Class – Example

- Adding the MyPrinter class extends PrintWriter in FilterWrapper class



```

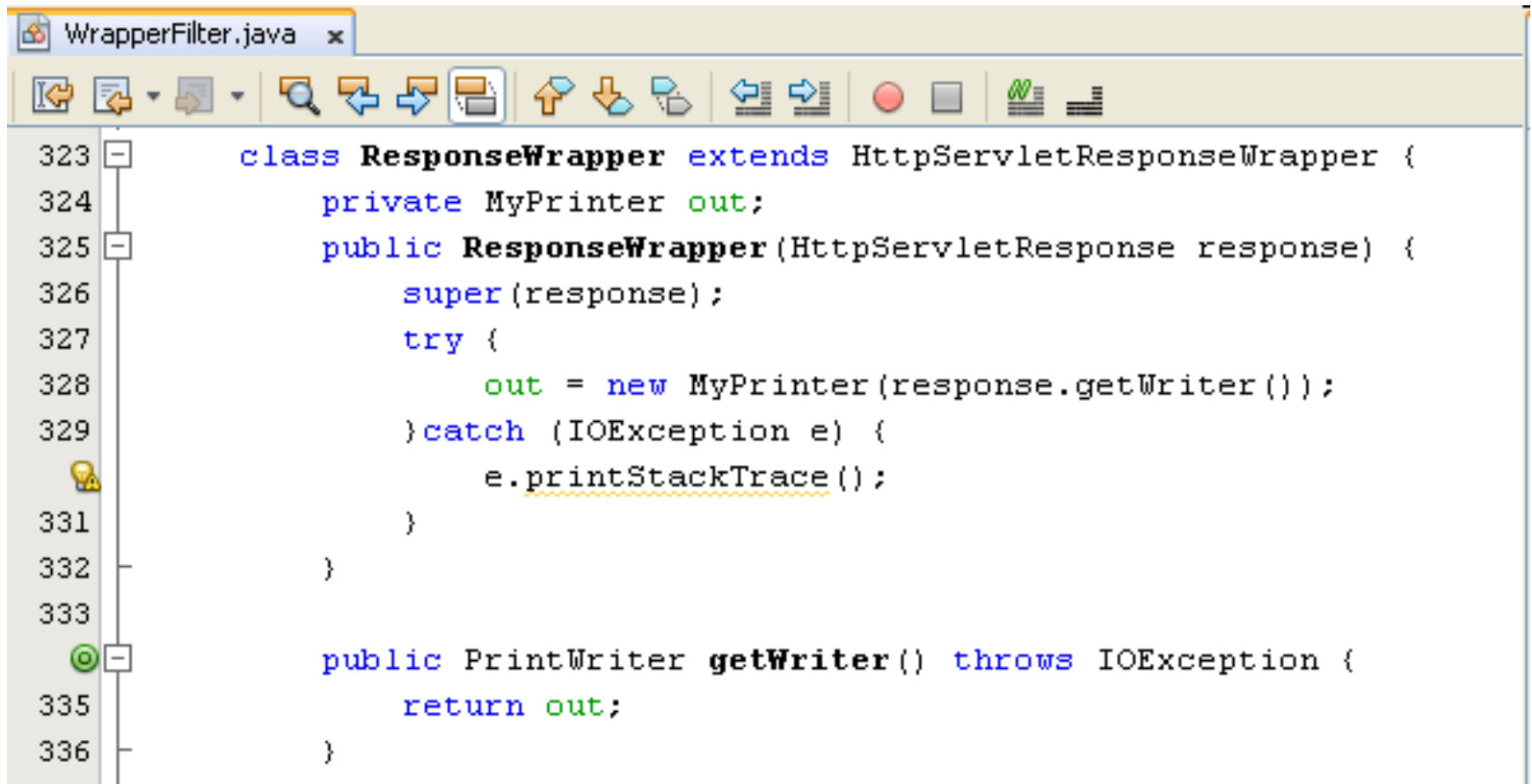
404      */
405    }
406    class MyPrinter extends PrintWriter {
407        public MyPrinter (Writer out) {
408            super(out);
409        }
410
411        public void close() {
412            super.close();
413        }
414    }
415 }
  
```



# Filter Chain

## Wrapper Class – Example

- Modifying the ResponseWrapper class uses MyPrinter to output stream



```

323 class ResponseWrapper extends HttpServletResponseWrapper {
324     private MyPrinter out;
325     public ResponseWrapper(HttpServletResponse response) {
326         super(response);
327         try {
328             out = new MyPrinter(response.getWriter());
329         } catch (IOException e) {
330             e.printStackTrace();
331         }
332     }
333
334     public PrintWriter getWriter() throws IOException {
335         return out;
336     }
  
```

# Filter Chain

## Wrapper Class – Example

```

WrapperFilter.java x
133  */
134  public void doFilter(ServletRequest request, ServletResponse response,
135                      FilterChain chain)
136      throws IOException, ServletException {
137      HttpServletResponse resp = (HttpServletResponse)response;
138      ResponseWrapper wrapperResp = new ResponseWrapper(resp);
139      try {
140          chain.doFilter(request, wrapperResp);
141          PrintWriter out = wrapperResp.getWriter();
142          out.println("<br/>The slide is licensed to KhanhKT");
143          out.close();
144      }
145      catch(Throwable t) {
146          t.printStackTrace();
147      }
148  }

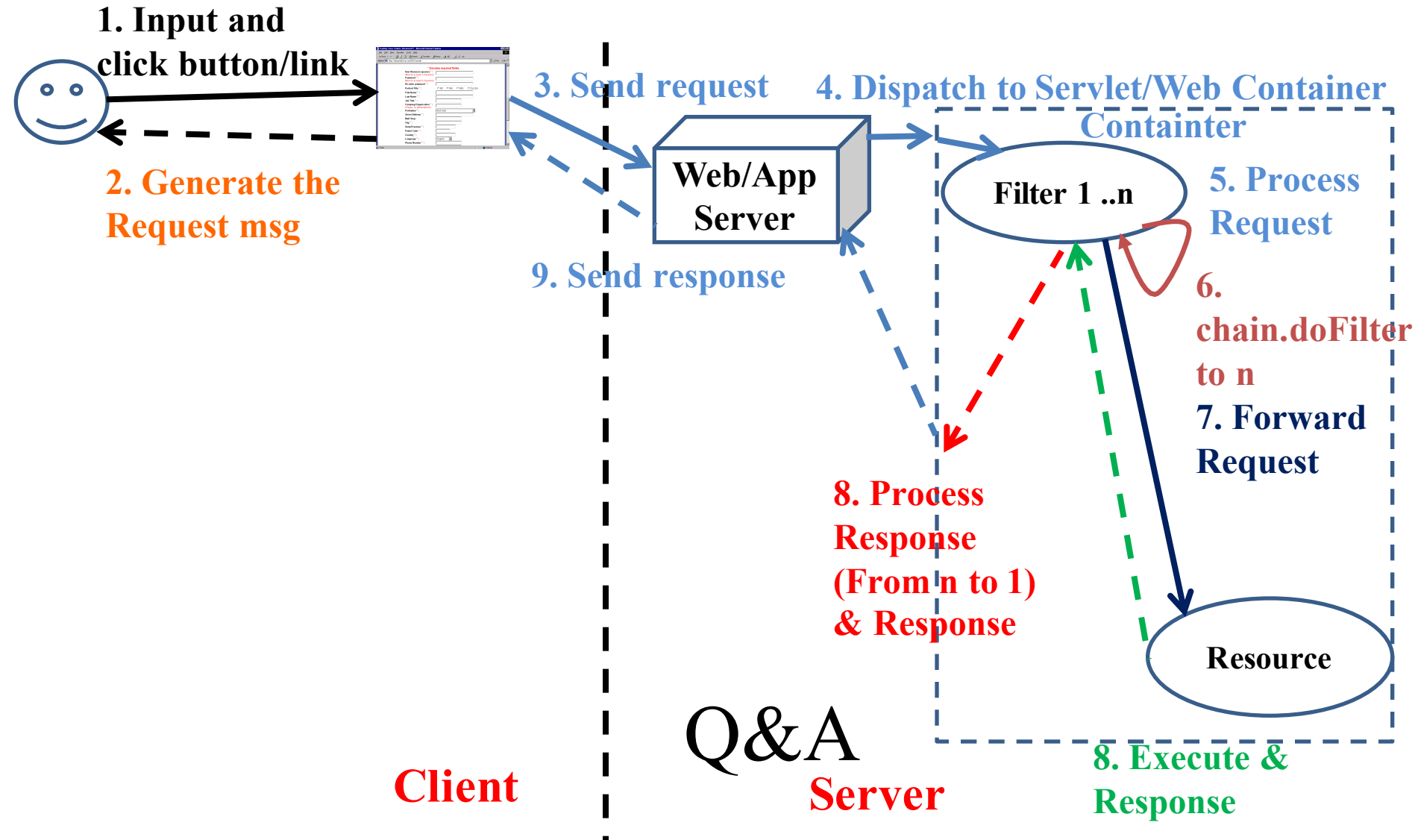
```

# MVC2

## Filter Acts as Controller

- While a **servlet** is the **most common controller**, a **filter** can **act as a controller** too
- While a **servlet** only **handles access** to the **dynamic** part of the application, a **filter** can **serve all** the **resources** in application, **including static ones**
- A filter as the controller **allows** to **block all requests** to the application, including request for static contents

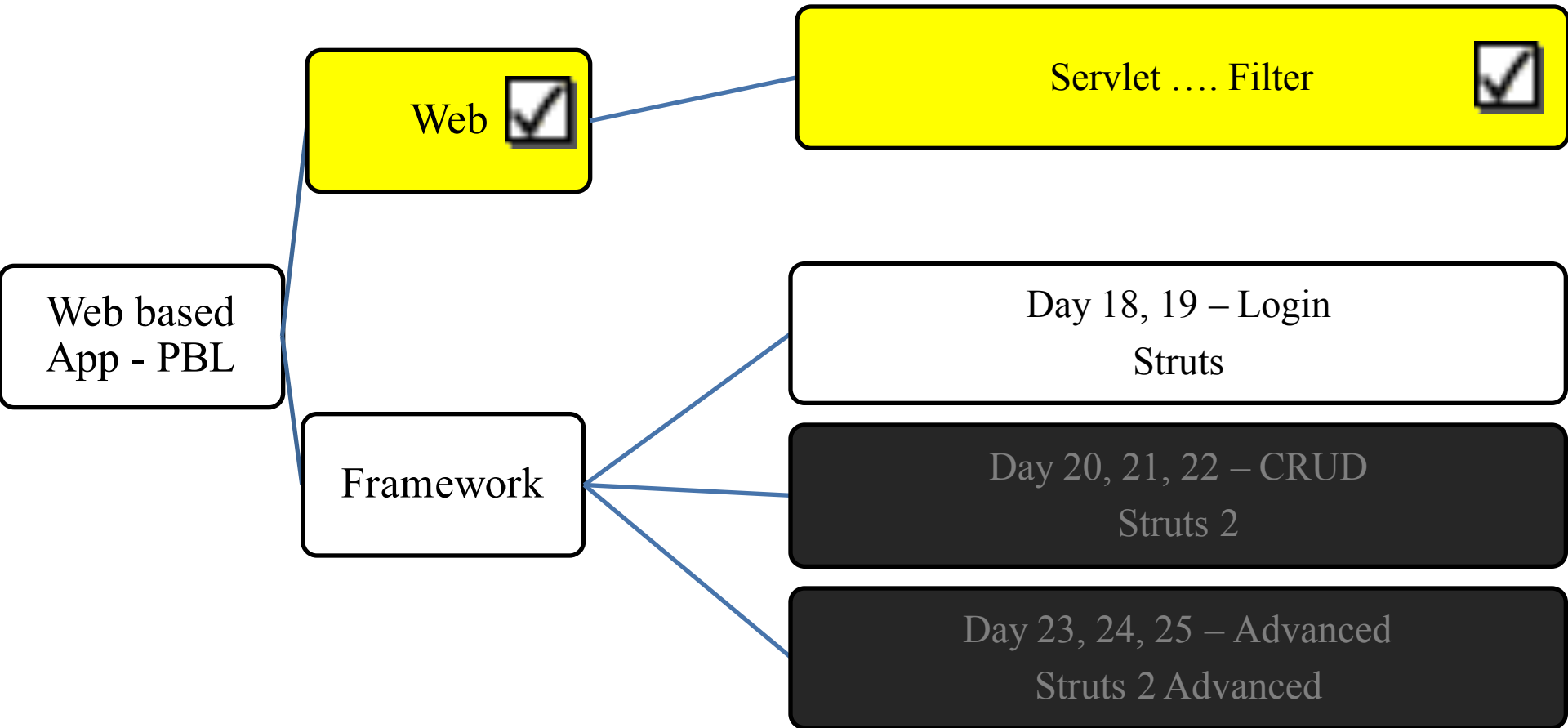
# Summary



# Next Lecture

- **How to build simple project MVC2 Web using Struts Framework?**
  - Struts
  - Struts Components
  - Struts Tag Lib (self-study)

# Next Lecture



# Appendix

## MVC2 using Filter as Controller

```

FilterDispatcher.java
Source History
23  * @author Kieu Trong Khanh
24  */
25  public class FilterDispatcher implements Filter {
26
27      private static final boolean debug = true;
28      private final String loginPage = "login.html";
29
30      // The filter configuration object we are associated with. If
31      // this value is null, this filter instance is not currently
32      // configured.
33      private FilterConfig filterConfig = null;
34
35      public FilterDispatcher() {...2 lines }
36
37
38      private void doBeforeProcessing(ServletRequest request, ServletResponse response)
39          throws IOException, ServletException {...26 lines }
40
41
42      private void doAfterProcessing(ServletRequest request, ServletResponse response)
43          throws IOException, ServletException {...23 lines }
44
45      /**...9 lines */
46      public void doFilter(ServletRequest request, ServletResponse response,
47                          FilterChain chain)
48          throws IOException, ServletException {
49          HttpServletRequest req = (HttpServletRequest) request;
50          String uri = req.getRequestURI();
51          String url = loginPage;
52
53      }
54
55      }
56
57      }
58
59      }
60
61      }
62
63      }
64
65      }
66
67      }
68
69      }
70
71      }
72
73      }
74
75      }
76
77      }
78
79      }
80
81      }
82
83      }
84
85      }
86
87      }
88
89      }
90
91      }
92
93      }
94
95      }
96
97      }
98
99      }
100
101      }
102
103      }
104
105      }
106
107      }
108
109      }
110
111      }
112
113      }
114
115      }
116
117      }
118
119      }
120
121      }
122
123      }
124
125      }
126
127      }
128
129      }
130
131      }
132
133      }
134
135      }
136
137      }
138
139      }
140
141      }
142
143      }
144
145      }
146
147      }
148
149      }
150
151      }
152
153      }
154
155      }
156
157      }
158
159      }
160
161      }
162
163      }
164
165      }
166
167      }
168
169      }
170
171      }
172
173      }
174
175      }
176
177      }
178
179      }
180
181      }
182
183      }
184
185      }
186
187      }
188
189      }
190
191      }
192
193      }
194
195      }
196
197      }
198
199      }
200
201      }
202
203      }
204
205      }
206
207      }
208
209      }
210
211      }
212
213      }
214
215      }
216
217      }
218
219      }
220
221      }
222
223      }
224
225      }
226
227      }
228
229      }
230
231      }
232
233      }
234
235      }
236
237      }
238
239      }
240
241      }
242
243      }
244
245      }
246
247      }
248
249      }
250
251      }
252
253      }
254
255      }
256
257      }
258
259      }
260
261      }
262
263      }
264
265      }
266
267      }
268
269      }
270
271      }
272
273      }
274
275      }
276
277      }
278
279      }
280
281      }
282
283      }
284
285      }
286
287      }
288
289      }
290
291      }
292
293      }
294
295      }
296
297      }
298
299      }
300
301      }
302
303      }
304
305      }
306
307      }
308
309      }
310
311      }
312
313      }
314
315      }
316
317      }
318
319      }
320
321      }
322
323      }
324
325      }
326
327      }
328
329      }
330
331      }
332
333      }
334
335      }
336
337      }
338
339      }
340
341      }
342
343      }
344
345      }
346
347      }
348
349      }
350
351      }
352
353      }
354
355      }
356
357      }
358
359      }
360
361      }
362
363      }
364
365      }
366
367      }
368
369      }
370
371      }
372
373      }
374
375      }
376
377      }
378
379      }
380
381      }
382
383      }
384
385      }
386
387      }
388
389      }
390
391      }
392
393      }
394
395      }
396
397      }
398
399      }
400
401      }
402
403      }
404
405      }
406
407      }
408
409      }
410
411      }
412
413      }
414
415      }
416
417      }
418
419      }
420
421      }
422
423      }
424
425      }
426
427      }
428
429      }
430
431      }
432
433      }
434
435      }
436
437      }
438
439      }
440
441      }
442
443      }
444
445      }
446
447      }
448
449      }
450
451      }
452
453      }
454
455      }
456
457      }
458
459      }
460
461      }
462
463      }
464
465      }
466
467      }
468
469      }
470
471      }
472
473      }
474
475      }
476
477      }
478
479      }
480
481      }
482
483      }
484
485      }
486
487      }
488
489      }
490
491      }
492
493      }
494
495      }
496
497      }
498
499      }
500
501      }
502
503      }
504
505      }
506
507      }
508
509      }
510
511      }
512
513      }
514
515      }
516
517      }
518
519      }
520
521      }
522
523      }
524
525      }
526
527      }
528
529      }
530
531      }
532
533      }
534
535      }
536
537      }
538
539      }
540
541      }
542
543      }
544
545      }
546
547      }
548
549      }
550
551      }
552
553      }
554
555      }
556
557      }
558
559      }
560
561      }
562
563      }
564
565      }
566
567      }
568
569      }
570
571      }
572
573      }
574
575      }
576
577      }
578
579      }
580
581      }
582
583      }
584
585      }
586
587      }
588
589      }
590
591      }
592
593      }
594
595      }
596
597      }
598
599      }
600
601      }
602
603      }
604
605      }
606
607      }
608
609      }
610
611      }
612
613      }
614
615      }
616
617      }
618
619      }
620
621      }
622
623      }
624
625      }
626
627      }
628
629      }
630
631      }
632
633      }
634
635      }
636
637      }
638
639      }
640
641      }
642
643      }
644
645      }
646
647      }
648
649      }
650
651      }
652
653      }
654
655      }
656
657      }
658
659      }
660
661      }
662
663      }
664
665      }
666
667      }
668
669      }
670
671      }
672
673      }
674
675      }
676
677      }
678
679      }
680
681      }
682
683      }
684
685      }
686
687      }
688
689      }
690
691      }
692
693      }
694
695      }
696
697      }
698
699      }
700
701      }
702
703      }
704
705      }
706
707      }
708
709      }
710
711      }
712
713      }
714
715      }
716
717      }
718
719      }
720
721      }
722
723      }
724
725      }
726
727      }
728
729      }
730
731      }
732
733      }
734
735      }
736
737      }
738
739      }
740
741      }
742
743      }
744
745      }
746
747      }
748
749      }
750
751      }
752
753      }
754
755      }
756
757      }
758
759      }
760
761      }
762
763      }
764
765      }
766
767      }
768
769      }
770
771      }
772
773      }
774
775      }
776
777      }
778
779      }
780
781      }
782
783      }
784
785      }
786
787      }
788
789      }
790
791      }
792
793      }
794
795      }
796
797      }
798
799      }
800
801      }
802
803      }
804
805      }
806
807      }
808
809      }
810
811      }
812
813      }
814
815      }
816
817      }
818
819      }
820
821      }
822
823      }
824
825      }
826
827      }
828
829      }
830
831      }
832
833      }
834
835      }
836
837      }
838
839      }
840
841      }
842
843      }
844
845      }
846
847      }
848
849      }
850
851      }
852
853      }
854
855      }
856
857      }
858
859      }
860
861      }
862
863      }
864
865      }
866
867      }
868
869      }
870
871      }
872
873      }
874
875      }
876
877      }
878
879      }
880
881      }
882
883      }
884
885      }
886
887      }
888
889      }
890
891      }
892
893      }
894
895      }
896
897      }
898
899      }
900
901      }
902
903      }
904
905      }
906
907      }
908
909      }
910
911      }
912
913      }
914
915      }
916
917      }
918
919      }
920
921      }
922
923      }
924
925      }
926
927      }
928
929      }
930
931      }
932
933      }
934
935      }
936
937      }
938
939      }
940
941      }
942
943      }
944
945      }
946
947      }
948
949      }
950
951      }
952
953      }
954
955      }
956
957      }
958
959      }
960
961      }
962
963      }
964
965      }
966
967      }
968
969      }
970
971      }
972
973      }
974
975      }
976
977      }
978
979      }
980
981      }
982
983      }
984
985      }
986
987      }
988
989      }
990
991      }
992
993      }
994
995      }
996
997      }
998
999      }
1000
1001      }
1002
1003      }
1004
1005      }
1006
1007      }
1008
1009      }
1010
1011      }
1012
1013      }
1014
1015      }
1016
1017      }
1018
1019      }
1020
1021      }
1022
1023      }
1024
1025      }
1026
1027      }
1028
1029      }
1030
1031      }
1032
1033      }
1034
1035      }
1036
1037      }
1038
1039      }
1040
1041      }
1042
1043      }
1044
1045      }
1046
1047      }
1048
1049      }
1050
1051      }
1052
1053      }
1054
1055      }
1056
1057      }
1058
1059      }
1060
1061      }
1062
1063      }
1064
1065      }
1066
1067      }
1068
1069      }
1070
1071      }
1072
1073      }
1074
1075      }
1076
1077      }
1078
1079      }
1080
1081      }
1082
1083      }
1084
1085      }
1086
1087      }
1088
1089      }
1090
1091      }
1092
1093      }
1094
1095      }
1096
1097      }
1098
1099      }
1100
1101      }
1102
1103      }
1104
1105      }
1106
1107      }
1108
1109      }
1110
1111      }
1112
1113      }
1114
1115      }
1116
1117      }
1118
1119      }
1120
1121      }
1122
1123      }
1124
1125      }
1126
1127      }
1128
1129      }
1130
1131      }
1132
1133      }
1134
1135      }
1136
1137      }
1138
1139      }
1140
1141      }
1142
1143      }
1144
1145      }
1146
1147      }
1148
1149      }
1150
1151      }
1152
1153      }
1154
1155      }
1156
1157      }
1158
1159      }
1160
1161      }
1162
1163      }
1164
1165      }
1166
1167      }
1168
1169      }
1170
1171      }
1172
1173      }
1174
1175      }
1176
1177      }
1178
1179      }
1180
1181      }
1182
1183      }
1184
1185      }
1186
1187      }
1188
1189      }
1190
1191      }
1192
1193      }
1194
1195      }
1196
1197      }
1198
1199      }
1200
1201      }
1202
1203      }
1204
1205      }
1206
1207      }
1208
1209      }
1210
1211      }
1212
1213      }
1214
1215      }
1216
1217      }
1218
1219      }
1220
1221      }
1222
1223      }
1224
1225      }
1226
1227      }
1228
1229      }
1230
1231      }
1232
1233      }
1234
1235      }
1236
1237      }
1238
1239      }
1240
1241      }
1242
1243      }
1244
1245      }
1246
1247      }
1248
1249      }
1250
1251      }
1252
1253      }
1254
1255      }
1256
1257      }
1258
1259      }
1260
1261      }
1262
1263      }
1264
1265      }
1266
1267      }
1268
1269      }
1270
1271      }
1272
1273      }
1274
1275      }
1276
1277      }
1278
1279      }
1280
1281      }
1282
1283      }
1284
1285      }
1286
1287      }
1288
1289      }
1290
1291      }
1292
1293      }
1294
1295      }
1296
1297      }
1298
1299      }
1300
1301      }
1302
1303      }
1304
1305      }
1306
1307      }
1308
1309      }
1310
1311      }
1312
1313      }
1314
1315      }
1316
1317      }
1318
1319      }
1320
1321      }
1322
1323      }
1324
1325      }
1326
1327      }
1328
1329      }
1330
1331      }
1332
1333      }
1334
1335      }
1336
1337      }
1338
1339      }
1340
1341      }
1342
1343      }
1344
1345      }
1346
1347      }
1348
1349      }
1350
1351      }
1352
1353      }
1354
1355      }
1356
1357      }
1358
1359      }
1360
1361      }
1362
1363      }
1364
1365      }
1366
1367      }
1368
1369      }
1370
1371      }
1372
1373      }
1374
1375      }
1376
1377      }
1378
1379      }
1380
1381      }
1382
1383      }
1384
1385      }
1386
1387      }
1388
1389      }
1390
1391      }
1392
1393      }
1394
1395      }
1396
1397      }
1398
1399      }
1400
1401      }
1402
1403      }
1404
1405      }
1406
1407      }
1408
1409      }
1410
1411      }
1412
1413      }
1414
1415      }
1416
1417      }
1418
1419      }
1420
1421      }
1422
1423      }
1424
1425      }
1426
1427      }
1428
1429      }
1430
1431      }
1432
1433      }
1434
1435      }
1436
1437      }
1438
1439      }
1440
1441      }
1442
1443      }
1444
1445      }
1446
1447      }
1448
1449      }
1450
1451      }
1452
1453      }
1454
1455      }
1456
1457      }
1458
1459      }
1460
1461      }
1462
1463      }
1464
1465      }
1466
1467      }
1468
1469      }
1470
1471      }
1472
1473      }
1474
1475      }
1476
1477      }
1478
1479      }
1480
1481      }
1482
1483      }
1484
1485      }
1486
1487      }
1488
1489      }
1490
1491      }
1492
1493      }
1494
1495      }
1496
1497      }
1498
1499      }
1500
1501      }
1502
1503      }
1504
1505      }
1506
1507      }
1508
1509      }
1510
1511      }
1512
1513      }
1514
1515      }
1516
1517      }
1518
1519      }
1520
1521      }
1522
1523      }
1524
1525      }
1526
1527      }
1528
1529      }
1530
1531      }
1532
1533      }
1534
1535      }
1536
1537      }
1538
1539      }
1540
1541      }
1542
1543      }
1544
1545      }
1546
1547      }
1548
1549      }
1550
1551      }
1552
1553      }
1554
1555      }
1556
1557      }
1558
1559      }
1560
1561      }
1562
1563      }
1564
1565      }
1566
1567      }
1568
1569      }
1570
1571      }
1572
1573      }
1574
1575      }
1576
1577      }
1578
1579      }
1580
1581      }
1582
1583      }
1584
1585      }
1586
1587      }
1588
1589      }
1590
1591      }
1592
1593      }
1594
1595      }
1596
1597      }
1598
1599      }
1600
1601      }
1602
1603      }
1604
1605      }
1606
1607      }
1608
1609      }
1610
1611      }
1612
1613      }
1614
1615      }
1616
1617      }
1618
1619      }
1620
1621      }
1622
1623      }
1624
1625      }
1626
1627      }
1628
1629      }
1630
1631      }
1632
1633      }
1634
1635      }
1636
1637      }
1638
1639      }
1640
1641      }
1642
1643      }
1644
1645      }
1646
1647      }
1648
1649      }
1650
1651      }
1652
1653      }
1654
1655      }
1656
1657      }
1658
1659      }
1660
1661      }
1662
1663      }
1664
1665      }
1666
1667      }
1668
1669      }
1670
1671      }
1672
1673      }
1674
1675      }
1676
1677      }
1678
1679      }
1680
1681      }
1682
1683      }
1684
1685      }
1686
1687      }
1688
1689      }
1690
1691      }
1692
1693      }
1694
1695      }
1696
1697      }
1698
1699      }
1700
1701      }
1702
1703      }
1704
1705      }
1706
1707      }
1708
1709      }
1710
1711      }
1712
1713      }
1714
1715      }
1716
1717      }
1718
1719      }
1720
1721      }
1722
1723      }
1724
1725      }
1726
1727      }
1728
1729      }
1730
1731      }
1732
1733      }
1734
1735      }
1736
1737      }
1738
1739      }
1740
1741      }
1742
1743      }
1744
1745      }
1746
1747      }
1748
1749      }
1750
1751      }
1752
1753      }
1754
1755      }
1756
1757      }
1758
1759      }
1760
1761      }
1762
1763      }
1764
1765      }
1766
1767      }
1768
1769      }
1770
1771      }
1772
1773      }
1774
1775      }
1776
1777      }
1778
1779      }
1780
1781      }
1782
1783      }
1784
1785      }
1786
1787      }
1788
1789      }
1790
1791      }
1792
1793      }
1794
1795      }
1796
1797      }
1798
1799      }
1800
1801      }
1802
1803      }
1804
1805      }
1806
1807      }
1808
1809      }
1810
1811      }
1812
1813      }
1814
1815      }
1816
1817      }
1818
1819      }
1820
1821      }
1822
1823      }
1824
1825      }
1826
1827      }
1828
1829      }
1830
1831      }
1832
1833      }
1834
1835      }
1836
1837      }
1838
1839      }
1840
1841      }
1842
1843      }
1844
1845      }
1846
1847      }
1848
1849      }
1850
1851      }
1852
1853      }
1854
1855      }
1856
1857      }
1858
1859      }
1860
1861      }
1862
1863      }
1864
1865      }
1866
1867      }
1868
1869      }
1870
1871      }
1872
1873      }
1874
1875      }
1876
1877      }
1878
1879      }
1880
1881      }
1882
1883      }
1884
1885      }
1886
1887      }
1888
1889      }
1890
1891      }
1892
1893      }
1894
1895      }
1896
1897      }
1898
1899      }
1900
1901      }
1902
1903      }
1904
1905      }
1906
1907      }
1908
1909      }
1910
1911      }
1912
1913      }
1914
1915      }
1916
1917      }
1918
1919      }
1920
1921      }
1922
1923      }
1924
1925      }
1926
1927      }
1928
1929      }
1930
1931      }
1932
1933      }
1934
1935      }
1936
1937      }
1938
1939      }
1940
1941      }
1942
1943      }
1944
1945      }
1946
1947      }
1948
1949      }
1950
1951      }
1952
1953      }
1954
1955      }
1956
1957      }
1958
1959      }
1960
1961      }
1962
1963      }
1964
1965      }
1966
1967      }
1968
1969      }
1970
1971      }
1972
1973      }
1974
1975      }
1976
1977      }
1978
1979      }
1980
1981      }
1982
1983      }
1984
1985      }
1986
1987      }
1988
1989      }
1990
1991      }
1992
1993      }
1994
1995      }
1996
1997      }
1998
1999      }
2000
2001      }
2002
2003      }
2004
2005      }
2006
2007      }
2008
2009      }
2010
2011      }
2012
2013      }
2014
2015      }
2016
2017      }
2018
2019      }
2020
2021      }
2022
2023      }
2024
2025      }
2026
2027      }
2028
2029      }
2030
2031      }
2032
2033      }
2034
2035      }
2036
2037      }
2038
2039      }
2040
2041      }
2042
2043      }
2044
2045      }
2046
2047      }
2048
2049      }
2050
2051      }
2052
2053      }
2054
2055      }
2056
2057      }
2058
2059      }
2060
2061      }
2062
2063      }
2064
2065      }
2066
2067      }
2068
2069      }
2070
2071      }
2072
2073      }
2074
2075      }
2076
2077      }
2078
2079      }
2080
2081      }
2082
2083      }
2084
2085      }
2086
2087      }
2088
2089      }
2090
2091      }
2092
2093      }
2094
2095      }
2096
2097      }
2098
2099      }
2100
2101      }
2102
2103      }
2104
2105      }
2106
2107      }
2108
2109      }
2110
2111      }
2112
2113      }
2114
2115      }
2116
2117      }
2118
2119      }
2120
2121      }
2122
2123      }
2124
2125      }
2126
2127      }
2128
2129      }
2130
2131      }
2132
2133      }
2134
2135      }
2136
2137      }
2138
2139      }
2140
2141      }
2142
2143      }
2144
2145      }
2146
2147      }
2148
2149      }
2150
2151      }
2152
2153      }
2154
2155      }
2156
2157      }
2158
2159      }
2160
2161      }
2162
2163      }
2164
2165      }
2166
2167      }
2168
2169      }
2170
2171      }
2172
2173      }
2174
2175      }
2176
2177      }
2178
2179      }
2180
2181      }
2182
2183      }
2184
2185      }
2186
2187      }
2188
2189      }
2190
2191      }
2192
2193      }
2194
2195      }
2196
2197      }
2198
2199      }
2200
2201      }
2202
2203      }
2
```

# Appendix

## MVC2 using Filter as Controller

```

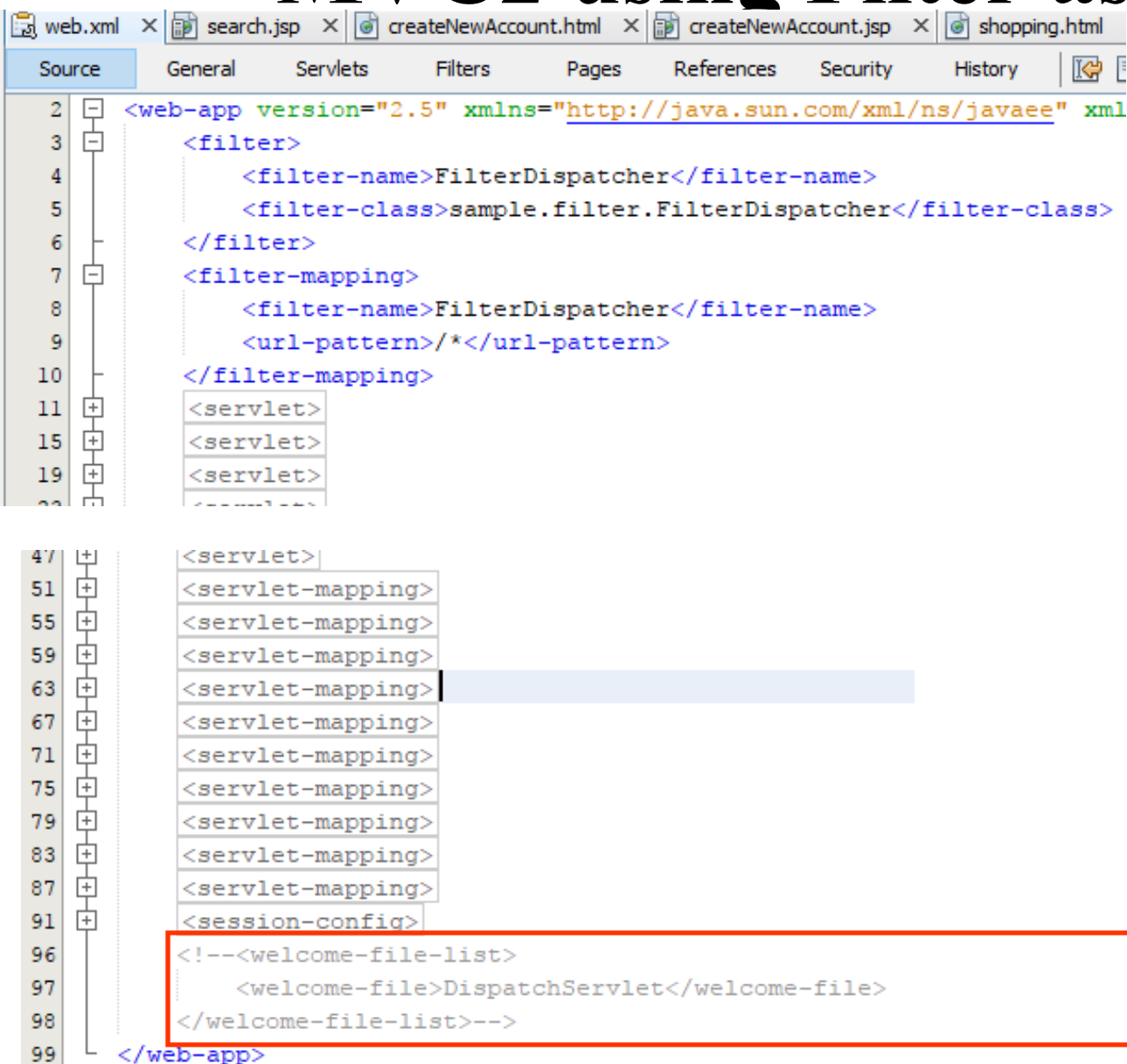
FilterDispatcher.java
Source History
107     try {
108         int lastIndex = uri.lastIndexOf("/");
109         String resource = uri.substring(lastIndex + 1);
110
111         if (resource.length() > 0) {
112             url = resource.substring(0, 1).toUpperCase()
113                 + resource.substring(1)
114                 + "Servlet";
115
116             if (resource.lastIndexOf(".html") > 0) {
117                 url = resource;
118             }
119         }
120
121         if (url != null) {
122             RequestDispatcher rd = req.getRequestDispatcher(url);
123             rd.forward(request, response);
124         } else {
125             chain.doFilter(request, response);
126         }
127     } catch (Throwable t) {
128         // If an exception is thrown somewhere down the filter chain,
129         // we still want to execute our after processing, and then
130         // rethrow the problem after that.
131         t.printStackTrace();
132     }
133 }

```



# Appendix

## MVC2 using Filter as Controller



```

web.xml x search.jsp x createNewAccount.html x createNewAccount.jsp x shopping.html
Source General Servlets Filters Pages References Security History
2 <web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee" xml
3 <filter>
4 <filter-name>FilterDispatcher</filter-name>
5 <filter-class>sample.filter.FilterDispatcher</filter-class>
6 </filter>
7 <filter-mapping>
8 <filter-name>FilterDispatcher</filter-name>
9 <url-pattern>/*</url-pattern>
10 </filter-mapping>
11 <servlet>
15 <servlet>
19 <servlet>
23 <servlet>
47 <servlet>
51 <servlet-mapping>
55 <servlet-mapping>
59 <servlet-mapping>
63 <servlet-mapping>
67 <servlet-mapping>
71 <servlet-mapping>
75 <servlet-mapping>
79 <servlet-mapping>
83 <servlet-mapping>
87 <servlet-mapping>
91 <session-config>
96 <!--<welcome-file-list>
97 <welcome-file>DispatchServlet</welcome-file>
98 </welcome-file-list>-->
99 </web-app>
  
```

# Appendix

## MVC2 using Filter as Controller

```

login.html x search.jsp x createNewAccount.html x createNewAccount.jsp x shopping.html x viewCart
Source History
1 <!DOCTYPE html>
2 ...5 lines
7 <html>
8   <head>
9     <title>Login</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <h1>Login Page</h1>
15    <form action="login" method="POST">
16      Username <input type="text" name="txtUsername" value="" /><br/>
17      Password <input type="password" name="txtPassword" value="" /><br/>
18      <input type="submit" value="Login" name="btAction" />
19      <input type="reset" value="Reset" />
20    </form><br/>
21    <a href="shopping.html">Click here to buy books</a><br/>
22    <a href="createNewAccount.html">Click here to Sign Up</a>
23  </body>
24 </html>

```

# Appendix

## MVC2 using Filter as Controller

```

search.jsp x createNewAccount.html x createNewAccount.jsp x shopping.html x viewCart.jsp x DeleteRecord.jsp
Source History
4      Author      : kieuokhanh
5      --%>
6
7      <%@page import="sample.registration.RegistrationDTO"%>
8      <%@page import="java.util.List"%>
9      <%@page contentType="text/html" pageEncoding="UTF-8"%>
10     <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
11     <!DOCTYPE html>
12     <html>
13     <head>
14         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
15         <title>Search</title>
16     </head>
17     <body>
18         <font color="red">
19             Welcome, ${sessionScope.USERNAME}
20         </font>
21         <h1>Search Page</h1>
22         <form action="search">
23             Search Value <input type="text" name="txtSearchValue" value="" /><br/>
24             <input type="submit" value="Search" name="btAction" />
25         </form>
26         <br/>

```

# Appendix

## MVC2 using Filter as Controller

search.jsp × createNewAccount.html × createNewAccount.jsp × shopping.html × viewCart.jsp × DeleteRecordServlet.java × UpdateRecords

Source History

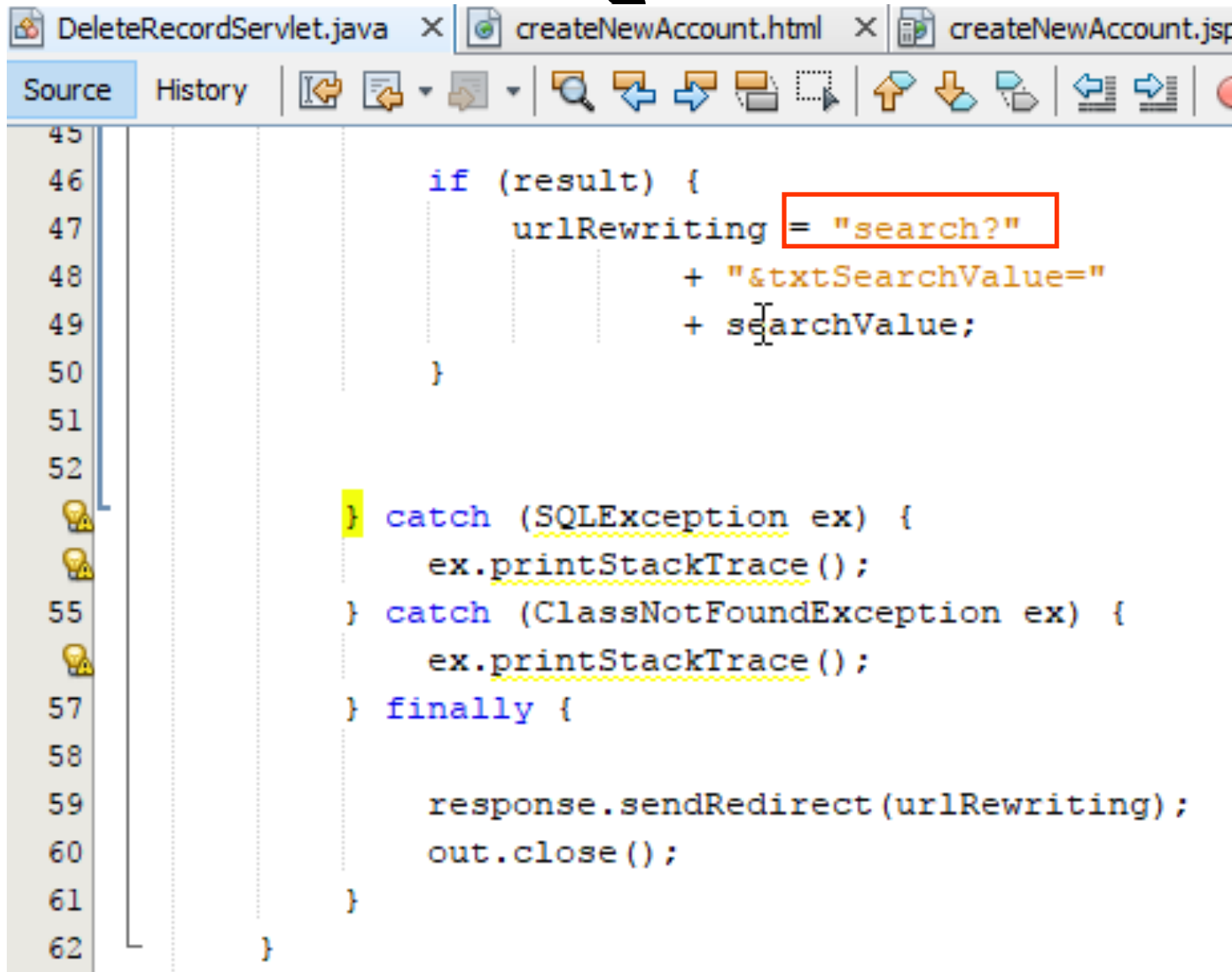
```

44 <tbody>
45 <c:forEach var="dto" items="${result}" varStatus="counter">
46 <form action="updateRecord">
47 <tr>
48 <...3 lines />
51 <...5 lines />
56 <...4 lines />
60 <...3 lines />
63 <...7 lines />
70 <td>
71 <c:url var="urlRewriting" value="deleteRecord">
72 <c:param name="pk" value="${dto.username}"/>
73 <c:param name="lastSearchValue" value="${param.txtSearchValue}"/>
74 </c:url>
75 <a href="${urlRewriting}">Delete</a>
76 </td>
77 <...5 lines />
82 </tr>
83 </form>
84 </c:forEach>
85 </tbody>
86 </table>

```

# Appendix

## MVC2 using Filter as Controller



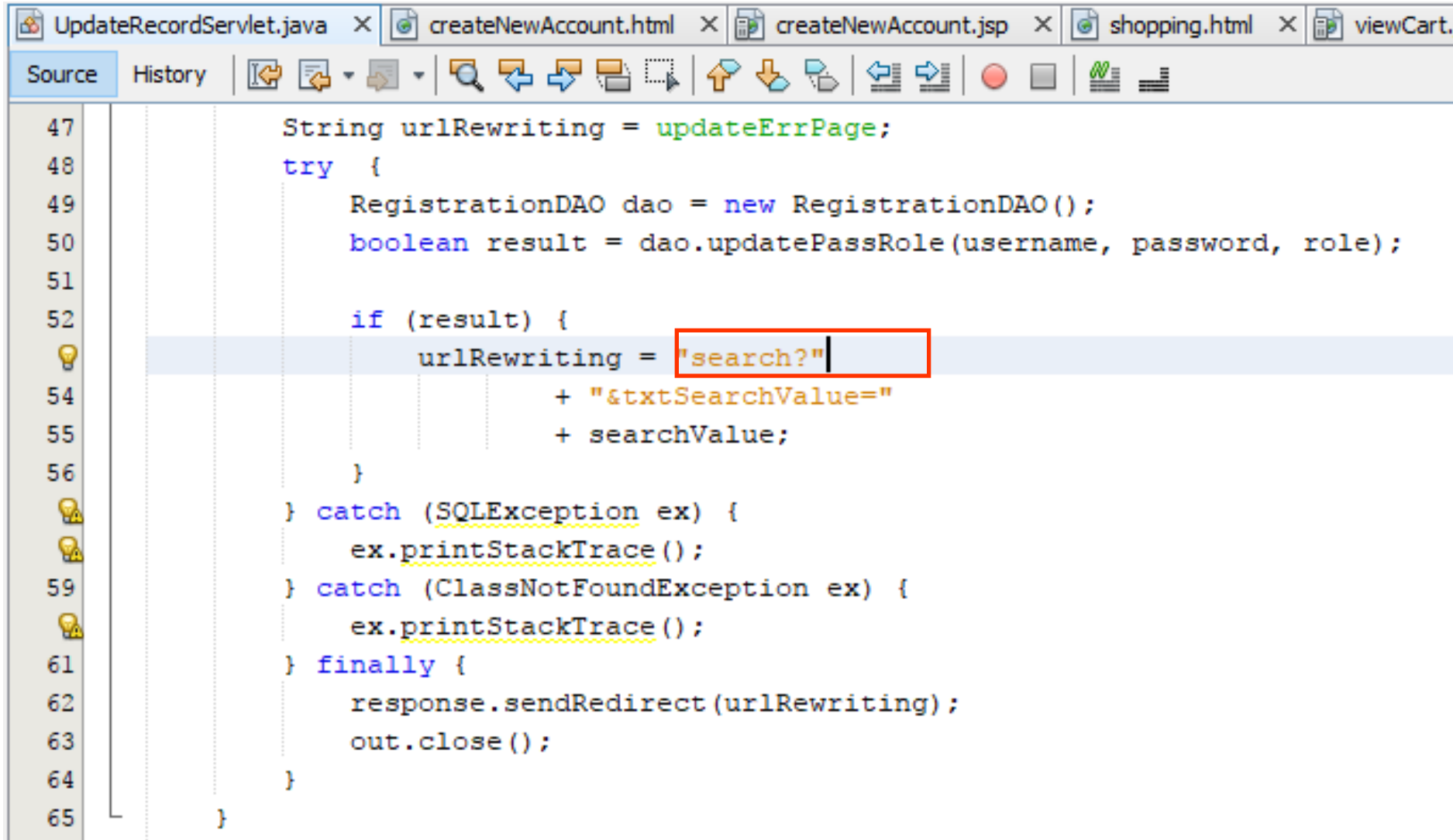
```

45
46         if (result) {
47             urlRewriting = "search?"
48                             + "&txtSearchValue="
49                             + searchValue;
50         }
51
52
53     } catch (SQLException ex) {
54         ex.printStackTrace();
55     } catch (ClassNotFoundException ex) {
56         ex.printStackTrace();
57     } finally {
58
59         response.sendRedirect(urlRewriting);
60         out.close();
61     }
62 }

```

# Appendix

## MVC2 using Filter as Controller



```

47      String urlRewriting = updateErrPage;
48      try {
49          RegistrationDAO dao = new RegistrationDAO();
50          boolean result = dao.updatePassRole(username, password, role);
51
52          if (result) {
53              urlRewriting = "search?"
54                          + "&txtSearchValue="
55                          + searchValue;
56          }
57      } catch (SQLException ex) {
58          ex.printStackTrace();
59      } catch (ClassNotFoundException ex) {
60          ex.printStackTrace();
61      } finally {
62          response.sendRedirect(urlRewriting);
63          out.close();
64      }
65  }
    
```

# Appendix

## MVC2 using Filter as Controller

```

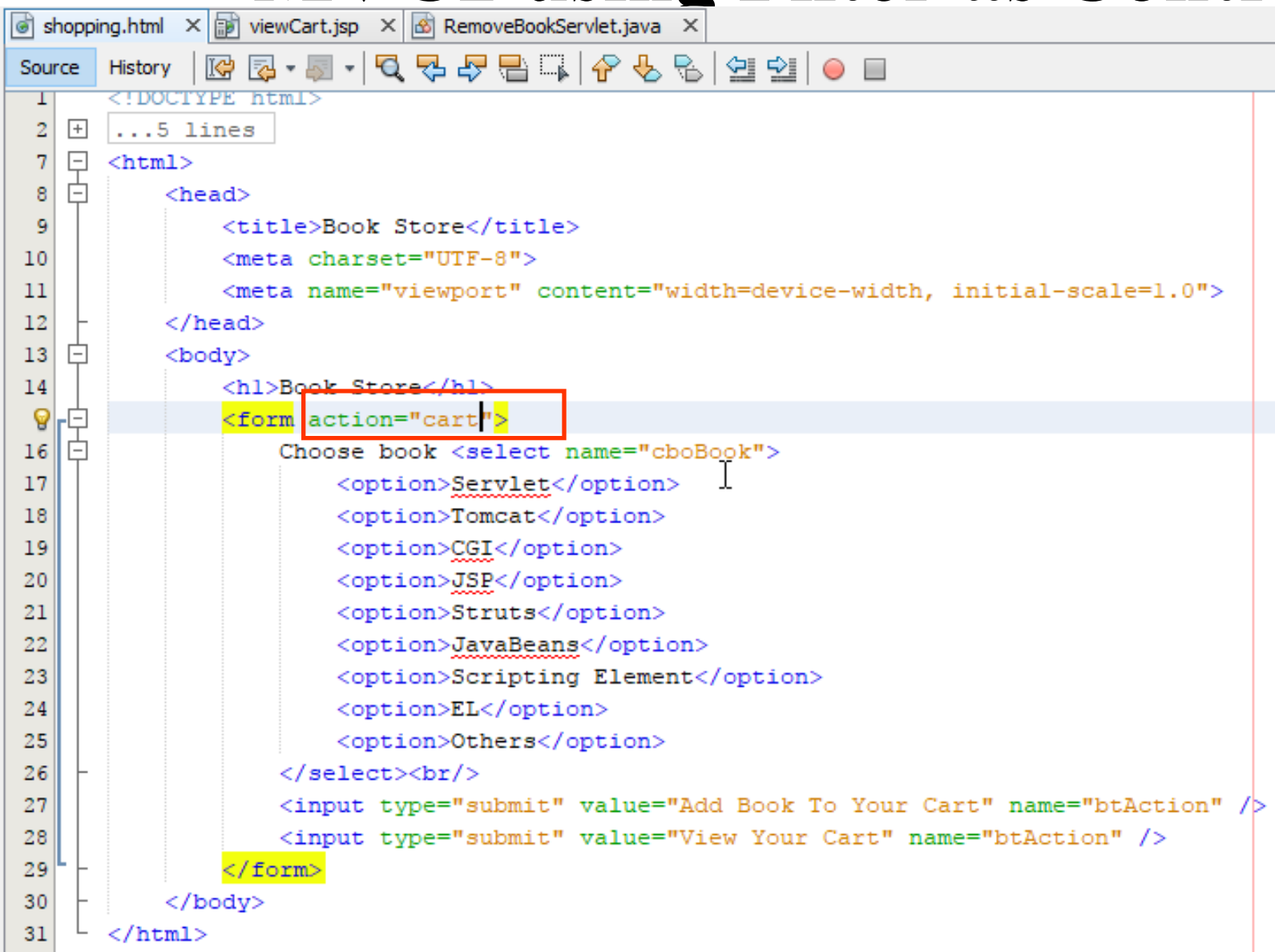
createNewAccount.html x createNewAccount.jsp x shopping.html x viewCart.jsp x RemoveBookServlet.java x
Source History
1 <!DOCTYPE html>
2 ...5 lines
7 <html>
8   <head>
9     <title>Create</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <h1>Create New Account</h1>
15    <form action="createRecord" method="POST">
16      Username* <input type="text" name="txtUsername" value="" />(6 - 20 chars)<br/>
17      Password* <input type="password" name="txtPassword" value="" />(6 - 30 chars)<br/>
18      Confirm <input type="password" name="txtConfirm" value="" /><br/>
19      Full name <input type="text" name="txtFullname" value="" />(2 - 50 chars)<br/>
20      <input type="submit" value="Create New Account" name="btAction" />
21      <input type="reset" value="Reset" />
22    </form>
23  </body>
24 </html>

```



# Appendix

## MVC2 using Filter as Controller



```

1  <!DOCTYPE html>
2  ...5 lines
7  <html>
8      <head>
9          <title>Book Store</title>
10         <meta charset="UTF-8">
11         <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     </head>
13     <body>
14         <h1>Book Store</h1>
15         <form action="cart">
16             Choose book <select name="cboBook">
17                 <option>Servlet</option>
18                 <option>Tomcat</option>
19                 <option>CGI</option>
20                 <option>JSP</option>
21                 <option>Struts</option>
22                 <option>JavaBeans</option>
23                 <option>Scripting Element</option>
24                 <option>EL</option>
25                 <option>Others</option>
26             </select><br/>
27             <input type="submit" value="Add Book To Your Cart" name="btAction" />
28             <input type="submit" value="View Your Cart" name="btAction" />
29         </form>
30     </body>
31 </html>
  
```



# Appendix

## MVC2 using Filter as Controller

CartServlet.java

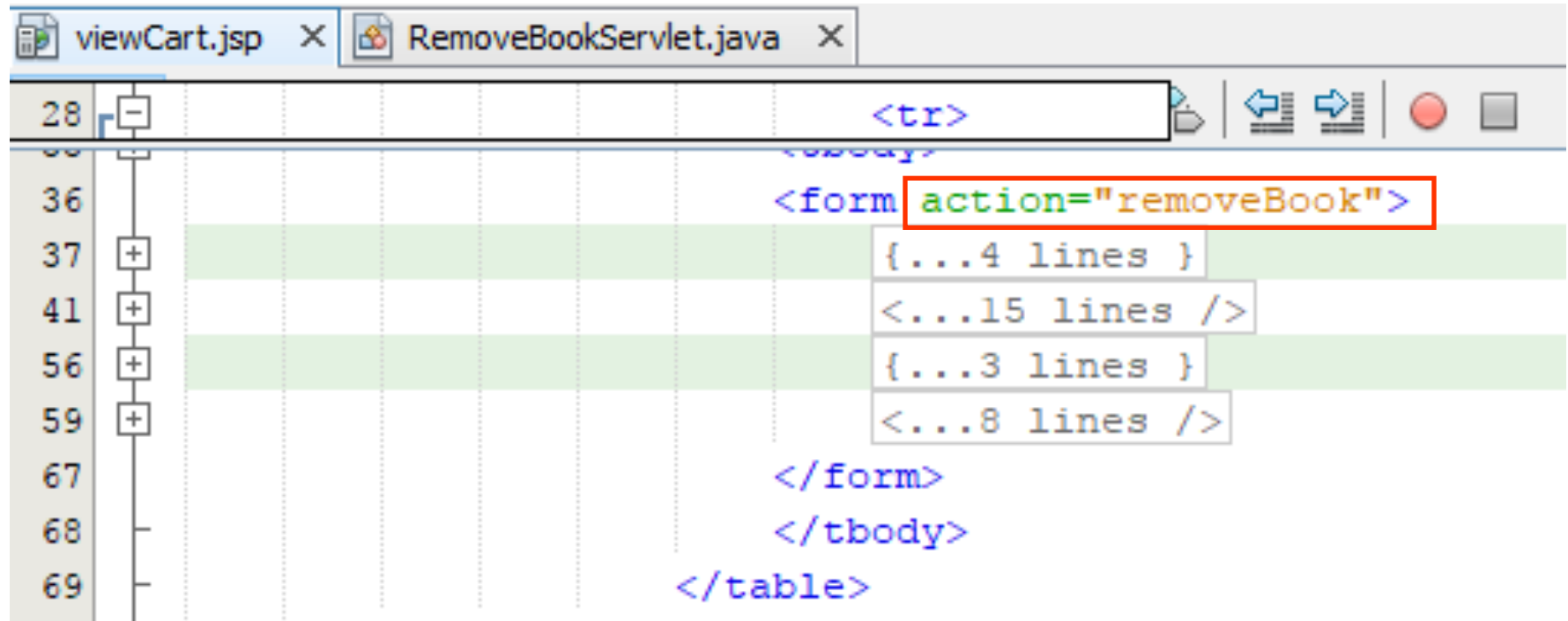
Source History

```

18  * @author Kieu Trong Khanh
19  */
20  public class CartServlet extends HttpServlet {
21
22      private final String addBookServlet = "AddBookServlet";
23      private final String viewCartPage = "viewCart.jsp";
24
25      /** Processes requests for both HTTP <code>GET</code> and <code>POST</code>
34  protected void processRequest(HttpServletRequest request, HttpServletResponse
35      throws ServletException, IOException {
36      response.setContentType("text/html;charset=UTF-8");
37      PrintWriter out = response.getWriter();
38
39      String action = request.getParameter("btAction");
40      String url = "";
41
42      try {
43          if (action.equals("Add Book To Your Cart")) {
44              url = addBookServlet;
45          } else if (action.equals("View Your Cart")) {
46              url = viewCartPage;
47          }
48      } finally {
49          RequestDispatcher rd = request.getRequestDispatcher(url);
50          rd.forward(request, response);
51          out.close();
52      }
53  }
  
```

# Appendix

## MVC2 using Filter as Controller



```

28  <tr>
36  <form action="removeBook">
37  { ... 4 lines }
41  <... 15 lines />
56  { ... 3 lines }
59  <... 8 lines />
67  </form>
68  </tbody>
69  </table>
  
```

# Appendix

## MVC2 using Filter as Controller

RemoveBookServlet.java

Source History

```

31
32     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
33         throws ServletException, IOException {
34         response.setContentType("text/html;charset=UTF-8");
35         PrintWriter out = response.getWriter();
36
37         String urlRewriting = "cart?btAction=View Your Cart";
38
39         try {
40             HttpSession session = request.getSession(false);
41
42             if (session != null) {
43                 CartObj cart = (CartObj)session.getAttribute("CART");
44
45                 if (cart != null) {

```

# Appendix

## MVC2 using Filter as Controller

### Projects

