Session 9, FILTERS

Objectives

Introduction to Filter

+ Understand the purpose of Filter

+ Create, Declaring and Mapping Filter

+ Ordering Filter Properly

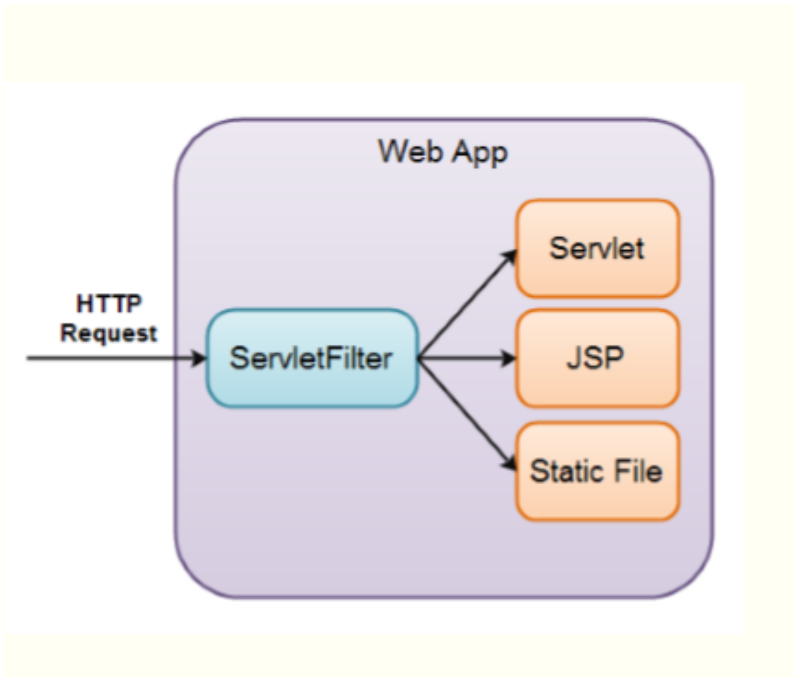
What is Filter?

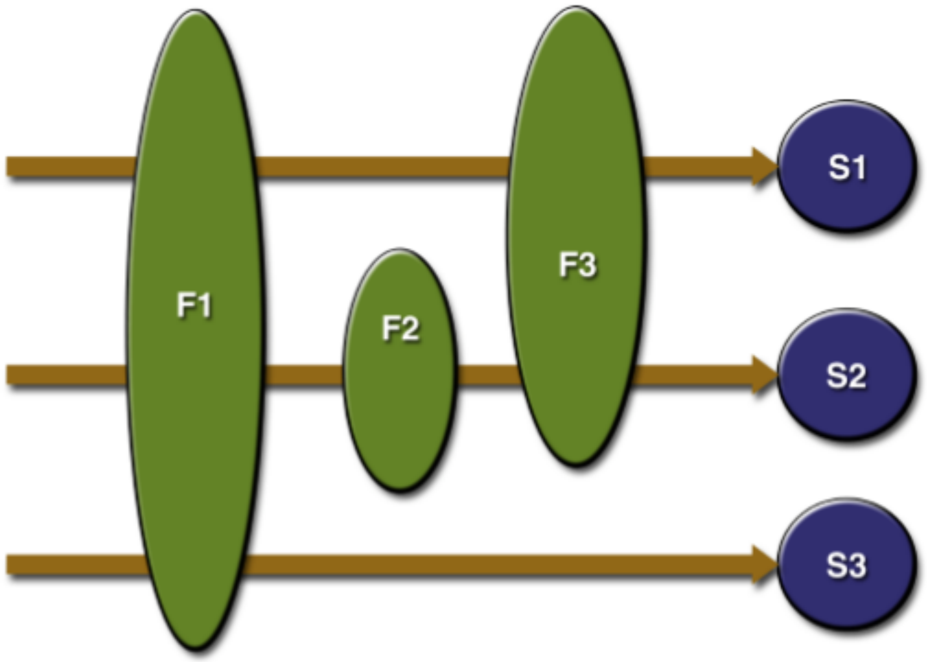
- Filter is a small program that run on the server before the servlet or JSP page with which it is associated.
- Filter can be attached to one or more servlets or JSP pages and can examine the request information going into these resources.
- Filter can Invoke the resource in normal manner.
- Filter can Invoke the resource with modified request information.
- Filter can Invoke the resource but modify response before sending it to the client.
- Filter can Prevent resource from being invoked and instead redirect to a different resource, return a particular status code, or generate replacement output.

What it can be used for ?

- Authentication blocking request based on user identity.
- Logging and auditing – Tracking uses of a web application.
- Image Conversion – Scaling maps and so on.
- Data Compression – Making downloads smaller.
- Localization – Targeting a request and response in particular locale.
- XSL/T Transformation of XML content

One or many filters





Creating a Basic Filter

1. Create a class that implements the Filter Interface. [init, doFilter , destroy]
2. Put the filtering behavior in the doFilter method.
3. Call the doFilter method of the FilterChain object.
4. Register the filter with the appropriate servlets and JSP page.
5. Disable the invoker servlet. **

Create a class that implements the Filter Interface

- Must implement javax.servlet.Filter
- public void init(FilterConfig config) throws ServletException
- public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws ServletException, IOException Executed each time servlet is invoked.
- FilterChain is used to invoke next filter that is associated with the servlet or jsp page, if no more filters it will invoke servlet or jsp page itself.
- public void destroy()

Register the Filter

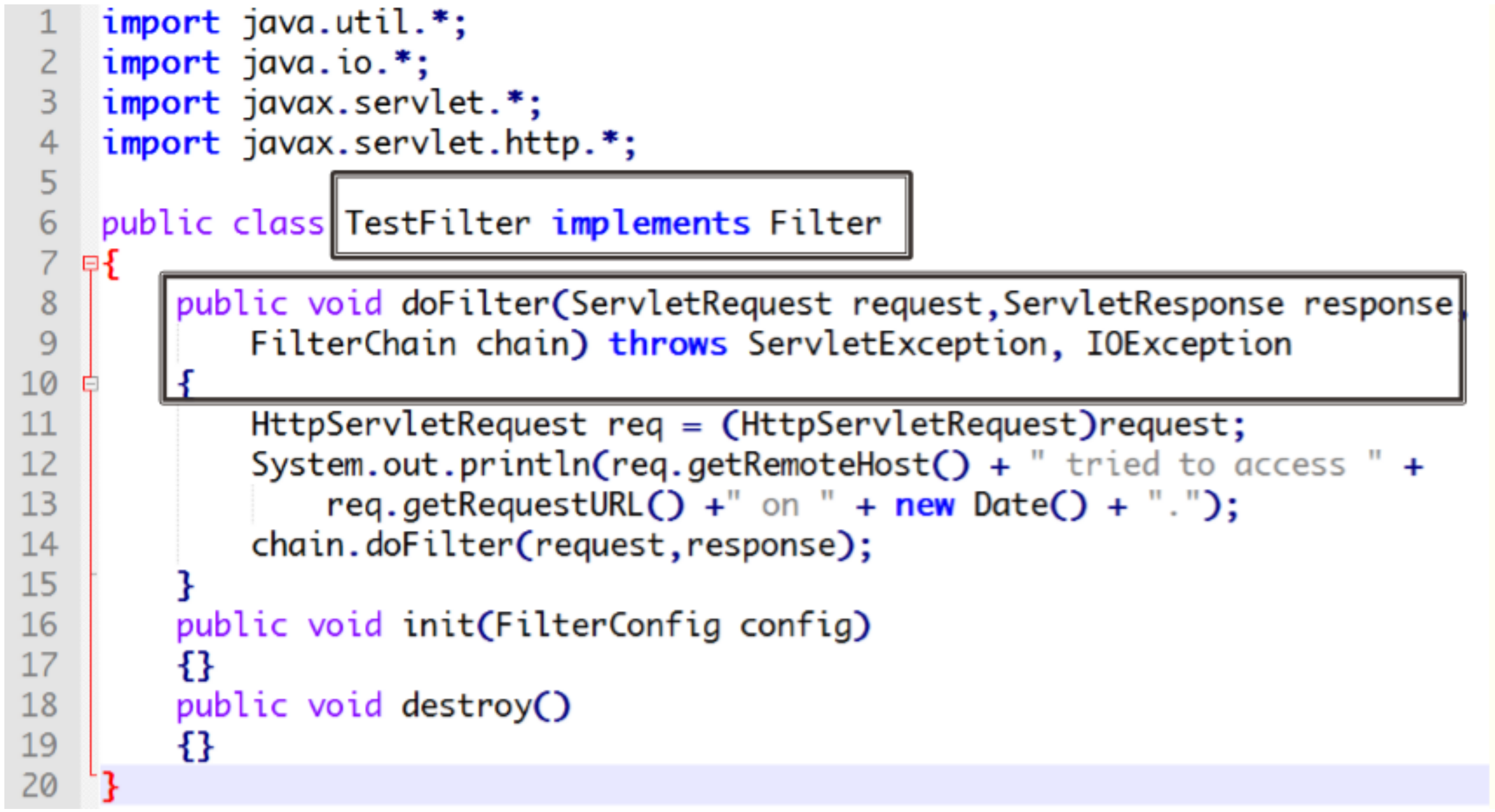
Use filter and filter-mapping tags to register filter in servlet.

```
<filter>

<filter-name> … </filter-name>

<filter-class> … </filter-class>

</filter>

<filter-mapping>

<filter-name> … </filter-name>

<url-pattern> … </url-pattern>

</filter-mapping>
```

Understanding filter and mapping tags

- In filter element we can have icon, filter-name, display-name, filter-class, init-param.
- In filter-mapping element filter-name, url-pattern, servlet-name, dispatcher.
- dispatcher : Optional used to specifies what type of request this filter-mapping should apply to.
- Possible values are REQUEST, FORWARD, INCLUDE and ERROR

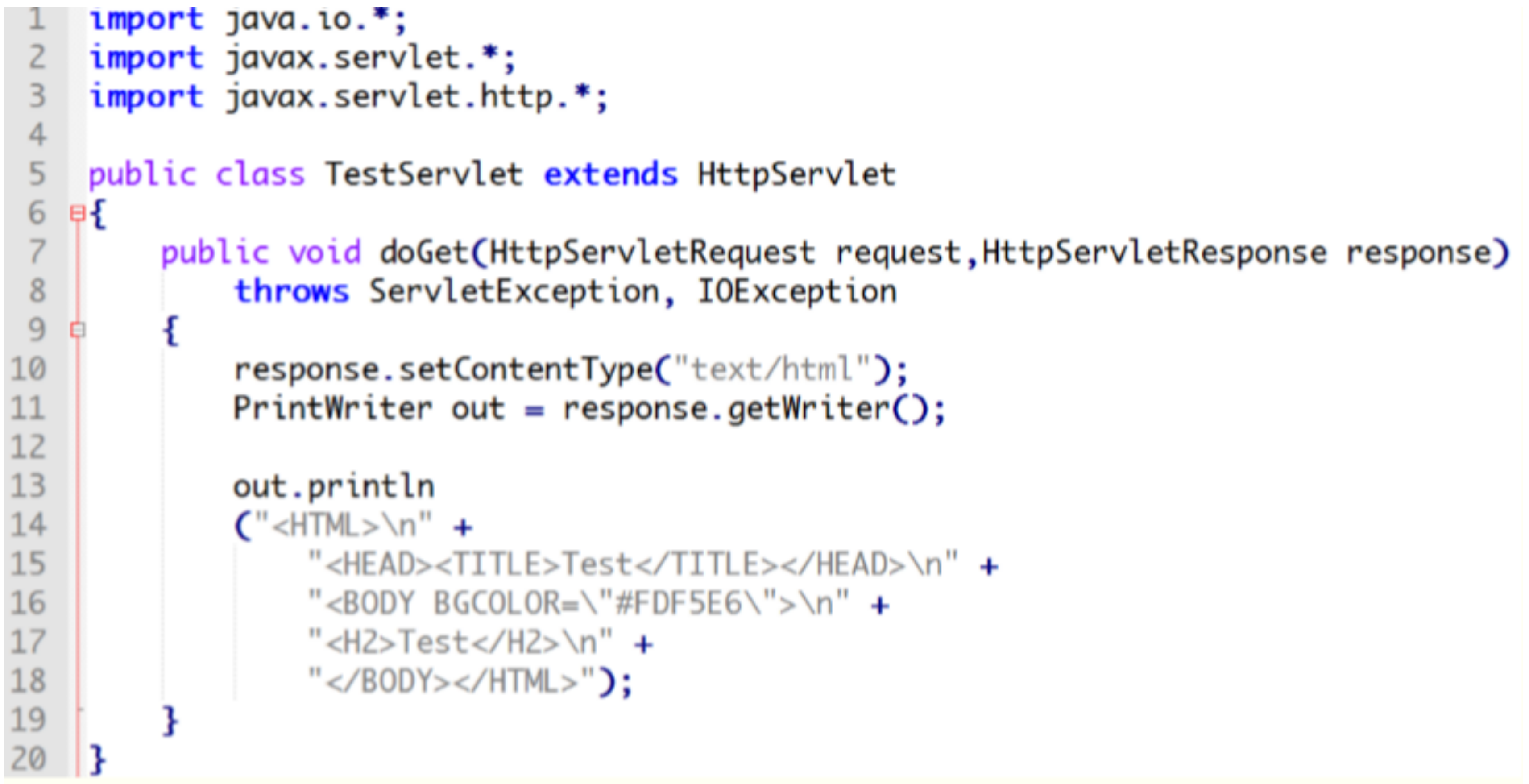
Example : Reporting Filter



```
import java.util.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class TestFilter implements Filter
{
    public void doFilter(ServletRequest request,ServletResponse response,
        FilterChain chain) throws ServletException, IOException
    {
        HttpServletRequest req = (HttpServletRequest)request;
        System.out.println(req.getRemoteHost() + " tried to access " +
            req.getRequestURL() +" on " + new Date() + ".");
        chain.doFilter(request,response);
    }
    public void init(FilterConfig config)
    {}
    public void destroy()
    {}
}
```

Example : Reporting Filter

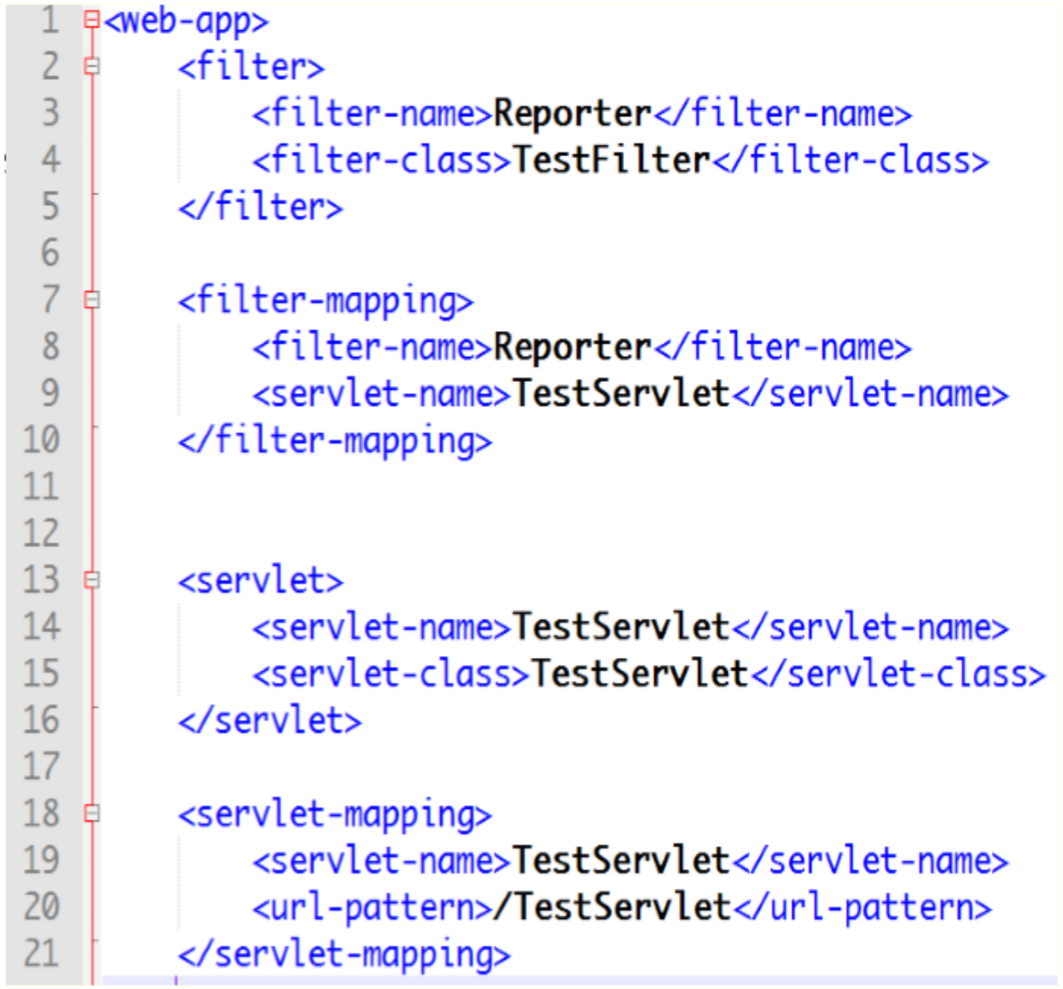
Code for Servlet



```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class TestServlet extends HttpServlet
{
    public void doGet(HttpServletRequest request,HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println
        ("<HTML>\n" +
            "<HEAD><TITLE>Test</TITLE></HEAD>\n" +
            "<BODY BGCOLOR=\"#FDF5E6\">\n" +
            "<H2>Test</H2>\n" +
            "</BODY></HTML>");
    }
}
```

Example : Reporting Filter

Web.xml

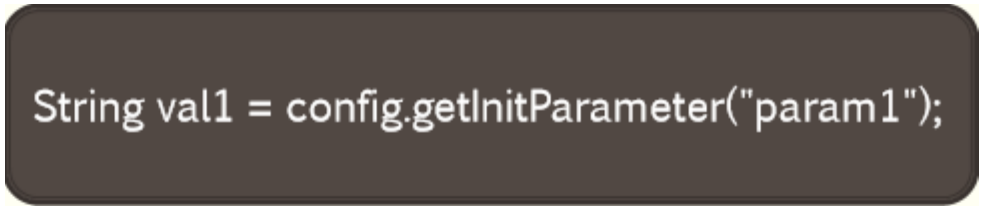


```xml
<web-app>
    <filter>
        <filter-name>Reporter</filter-name>
        <filter-class>TestFilter</filter-class>
    </filter>

    <filter-mapping>
        <filter-name>Reporter</filter-name>
        <servlet-name>TestServlet</servlet-name>
    </filter-mapping>


    <servlet>
        <servlet-name>TestServlet</servlet-name>
        <servlet-class>TestServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>TestServlet</servlet-name>
        <url-pattern>/TestServlet</url-pattern>
    </servlet-mapping>
```

Initialization Parameter in Filter

Developers, End Users, Deployers

```
<filter>
<filter-name>SomeFilter</filter-name>
<filter-class>somePackage.SomeFilterClass</filter-class>
<init-param>
<param-name>param1</param-name>
<param-value>value1</param-value>
</init-param>
</filter>
```



```
String val1 = config.getInitParameter("param1");
```

Blocking the Response (1)

What we do in a filter

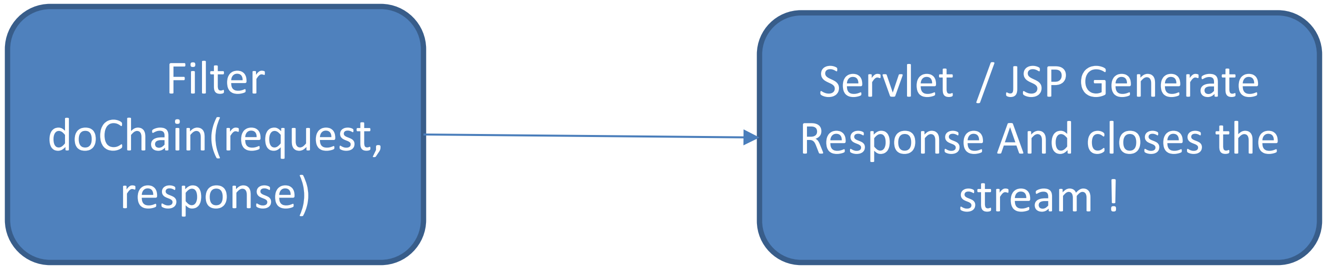
```
public void doFilter(ServletRequest request,ServletResponse response,
FilterChain chain) throws ServletException, IOException {
        HttpServletRequest req = (HttpServletRequest)request;
        context.log(req.getRemoteHost() + " tried to access " +
        req.getRequestURL() +" on " + new Date() + ".");
        chain.doFilter(request,response);
}
```

This will call next filter or servlet or jsp

Blocking the Response (2)

```
public void doFilter(ServletRequest request,ServletResponse response,
FilterChain chain) throws ServletException, IOException {
        HttpServletRequest req = (HttpServletRequest)request;
        context.log(req.getRemoteHost() + " tried to access " +
        req.getRequestURL() +" on " + new Date() + ".");
        if(condition==true)
                response.sendRedirect(“some other page /site”);
        else
                chain.doFilter(request,response);
}
```

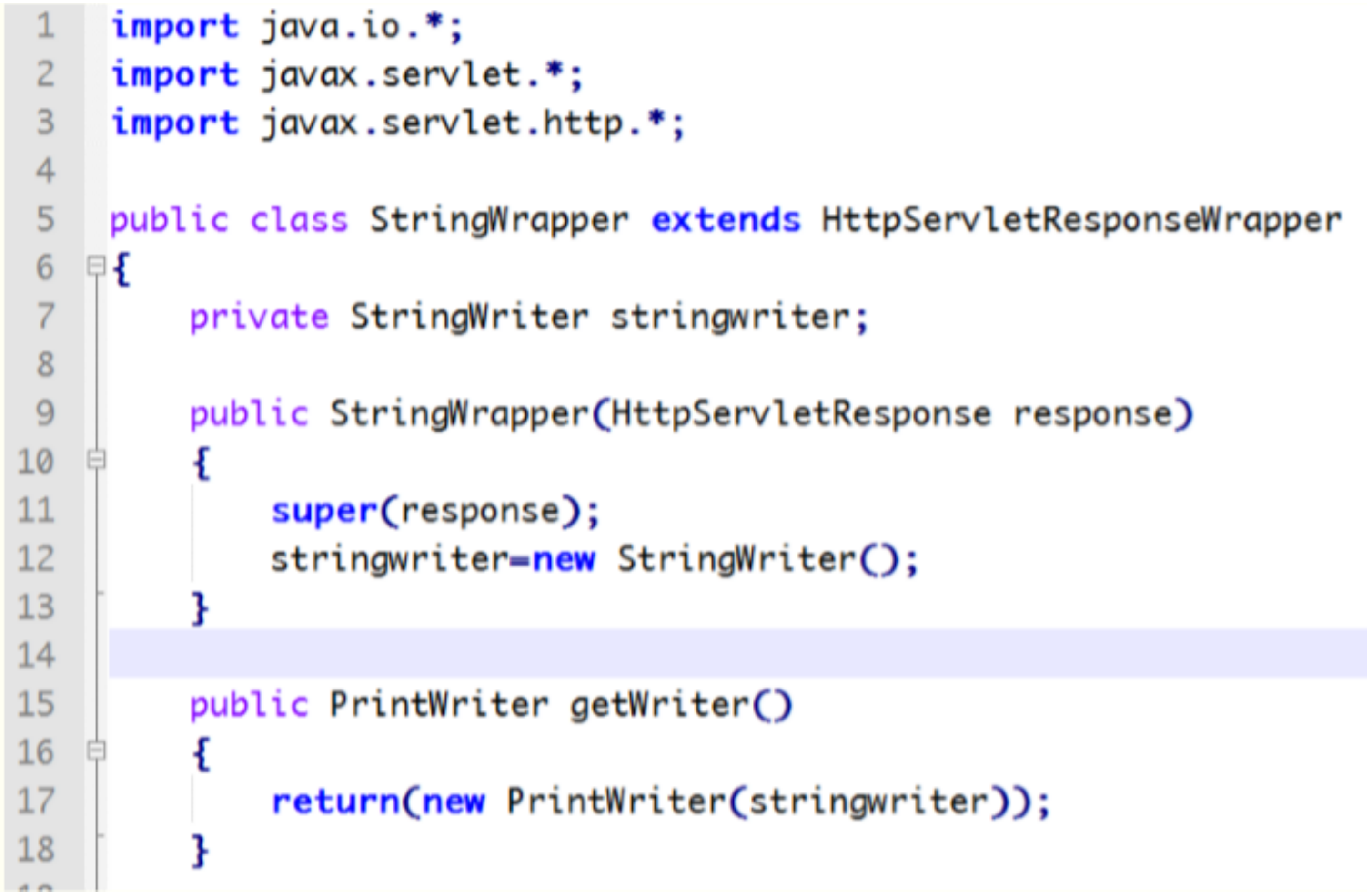
Modifying the response


Filter doChain(request, response)
Servlet / JSP Generate Response And closes the stream !

We will create our own space / buffer and give it to servlet and can manipulate the same once Servlet / JSP is done with response

Modifying the response:

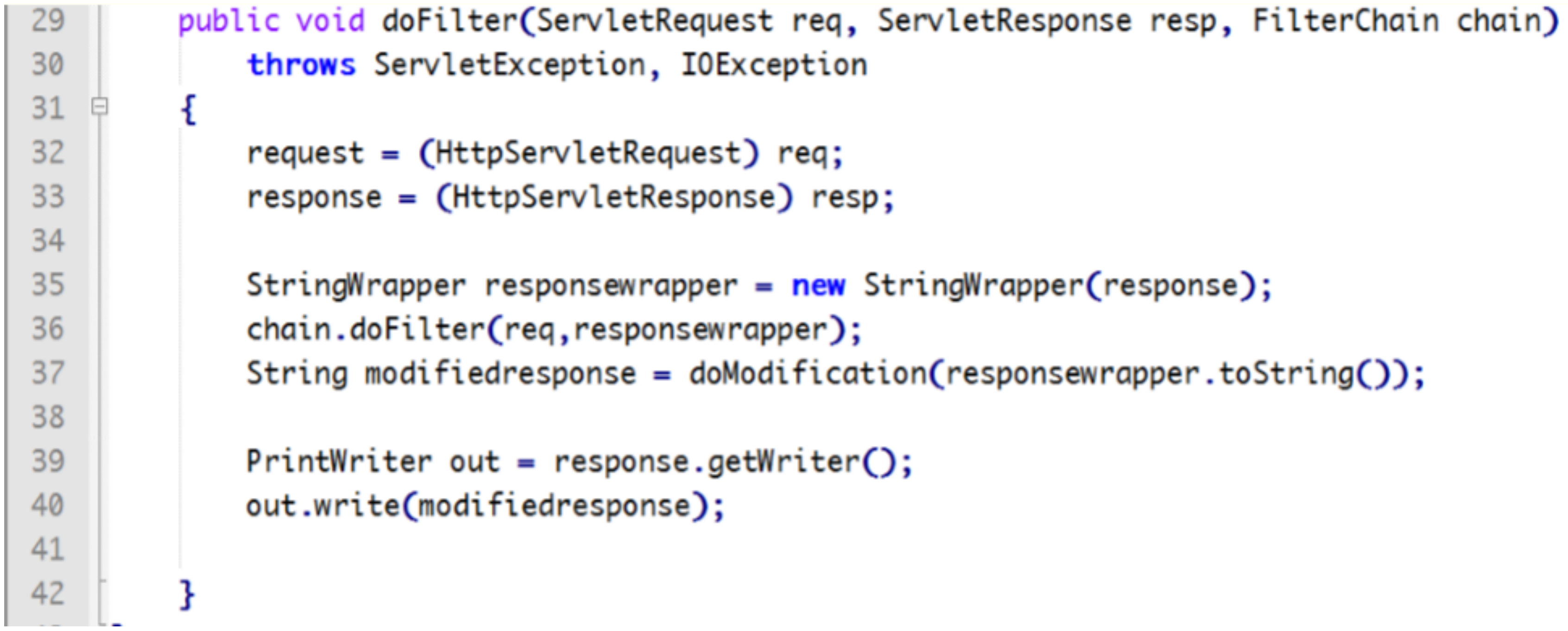
Creating our Wrapper Class



```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class StringWrapper extends HttpServletResponseWrapper
{
    private StringWriter stringwriter;

    public StringWrapper(HttpServletResponse response)
    {
        super(response);
        stringwriter=new StringWriter();
    }

    public PrintWriter getWriter()
    {
        return(new PrintWriter(stringwriter));
    }
```

Modifying the response:

In Filter Class



```java
public void doFilter(ServletRequest req, ServletResponse resp, FilterChain chain)
    throws ServletException, IOException
{
    request = (HttpServletRequest) req;
    response = (HttpServletResponse) resp;

    StringWrapper responsewrapper = new StringWrapper(response);
    chain.doFilter(req,responsewrapper);
    String modifiedresponse = doModification(responsewrapper.toString());

    PrintWriter out = response.getWriter();
    out.write(modifiedresponse);

}
```

Modifying the response:

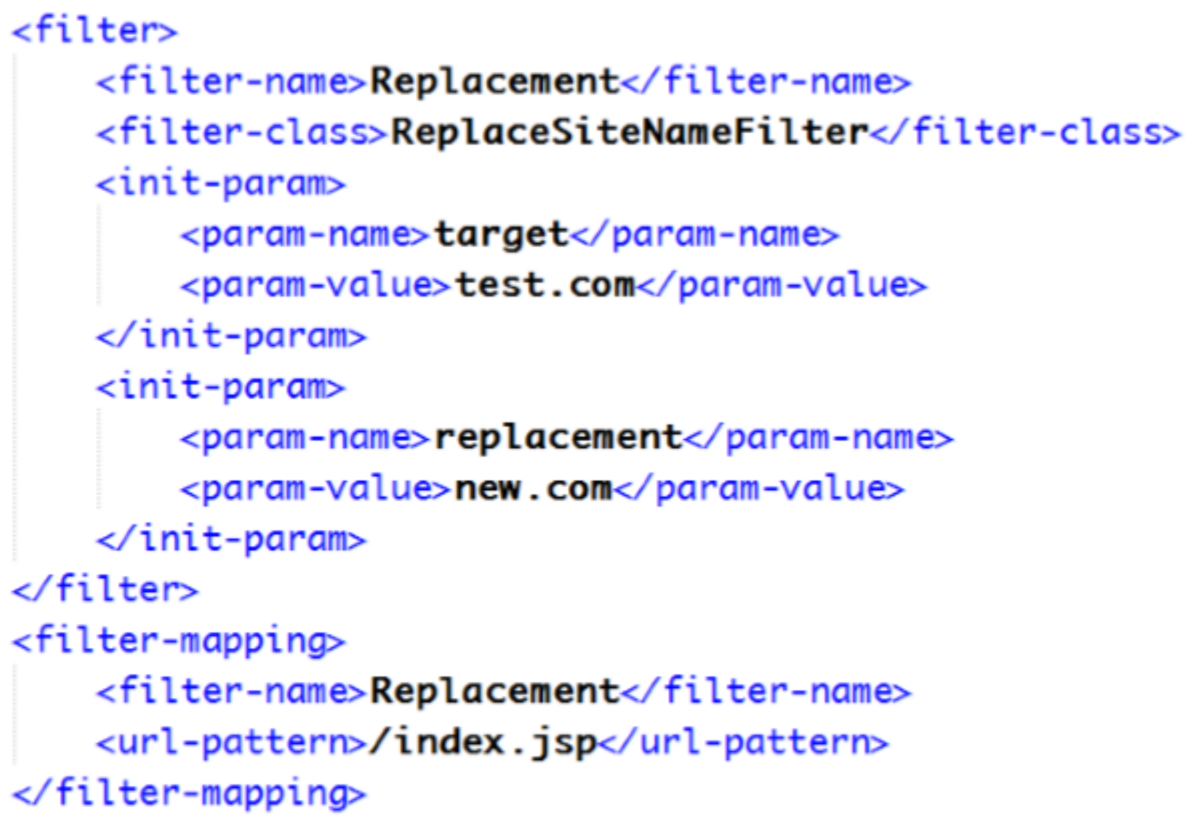
In Web.xml

Target will be replace

by replacement

Test.com will be

replace by new.com



```
<filter>
    <filter-name>Replacement</filter-name>
    <filter-class>ReplaceSiteNameFilter</filter-class>
    <init-param>
        <param-name>target</param-name>
        <param-value>test.com</param-value>
    </init-param>
    <init-param>
        <param-name>replacement</param-name>
        <param-value>new.com</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>Replacement</filter-name>
    <url-pattern>/index.jsp</url-pattern>
</filter-mapping>
```

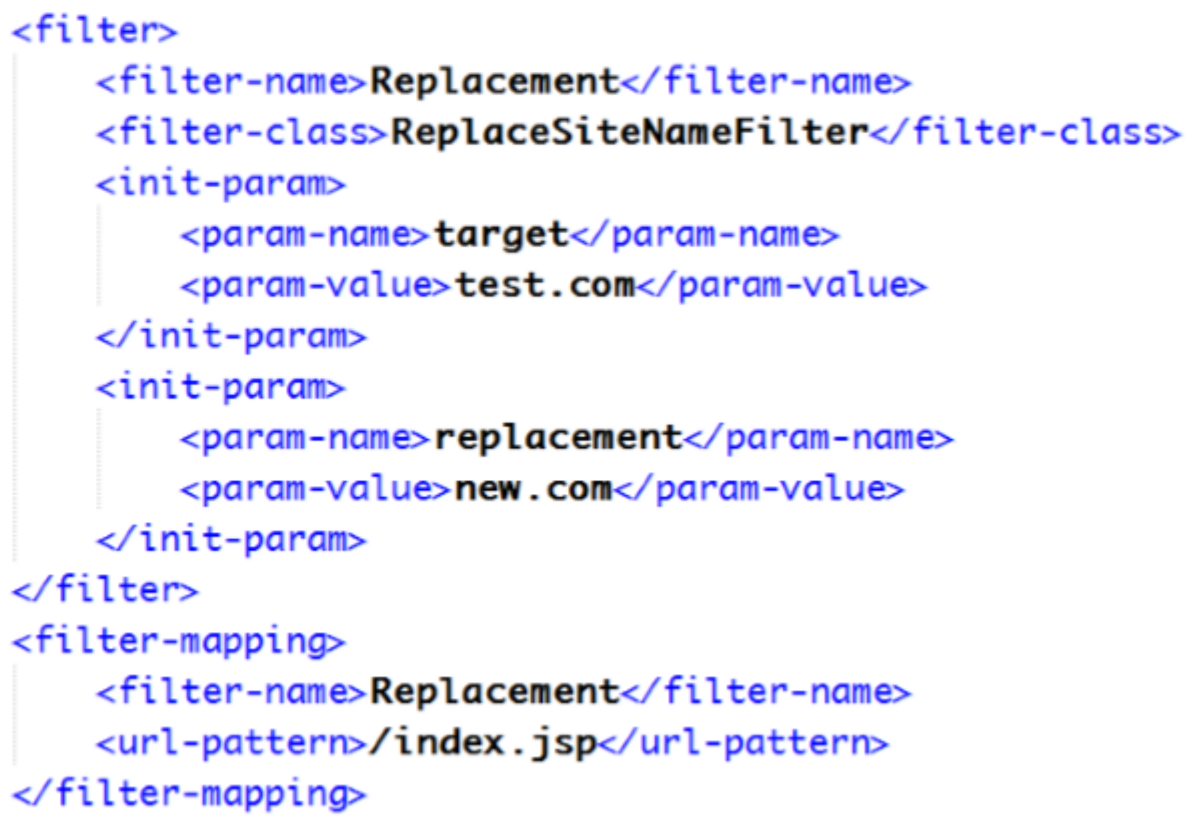
Modifying the response: Modification Method

Target will be replace

by replacement

Test.com will be

replace by new.com



```xml
<filter>
    <filter-name>Replacement</filter-name>
    <filter-class>ReplaceSiteNameFilter</filter-class>
    <init-param>
        <param-name>target</param-name>
        <param-value>test.com</param-value>
    </init-param>
    <init-param>
        <param-name>replacement</param-name>
        <param-value>new.com</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>Replacement</filter-name>
    <url-pattern>/index.jsp</url-pattern>
</filter-mapping>
```

Summary

1. **What's Filter?**
2. **What it can be used for ?**
3. **How to use Filter?**
   - Create a basic filter
   - Example
   - Initialization Parameter in Filter
   - Blocking the Response
   - Modifying the response