12장. 필터(Filter) - 로그

Filter

필터(Filter)

필터(Filter)

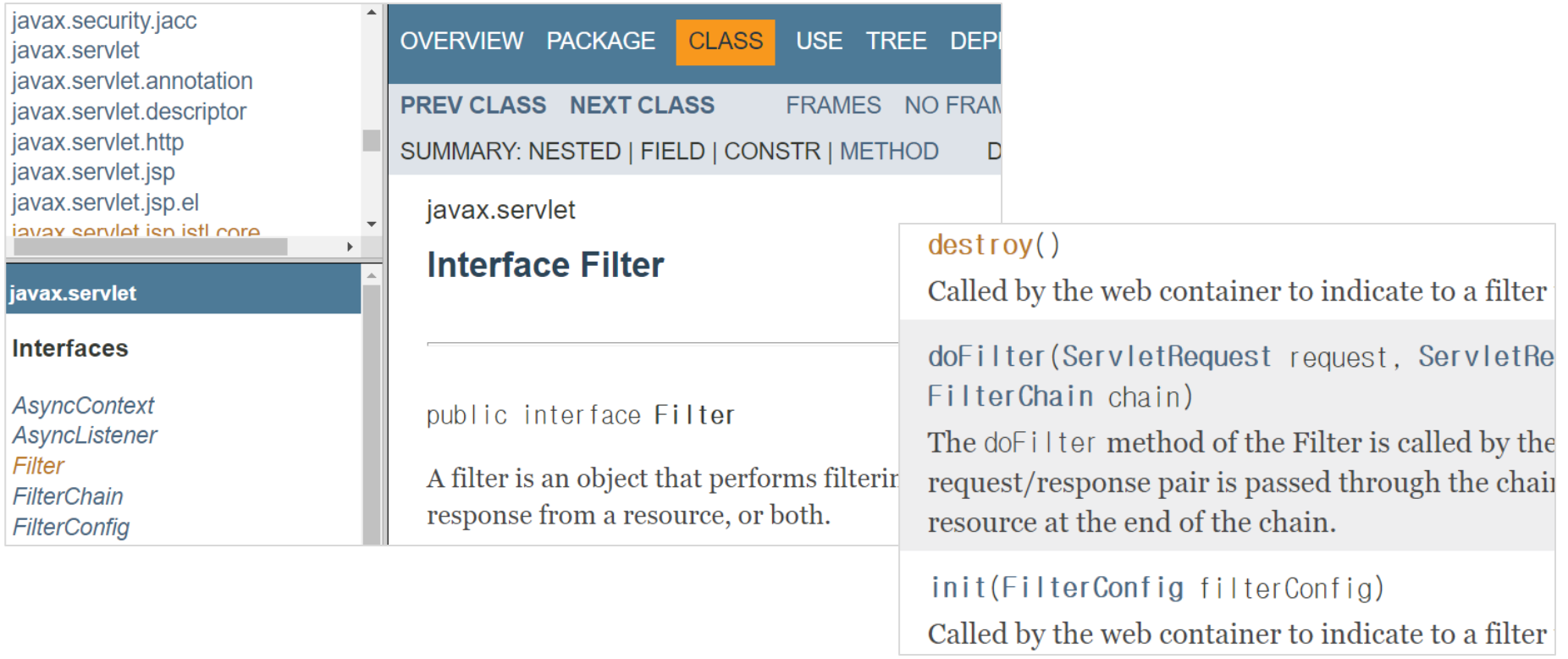
클라이언트와 서버 사이에서 request와 response 객체를 먼저 받아 사전/사후 작업 등 공통적으로 필요한 부분을 처리하는 것을 말한다.

웹 애플리케이션에 필터 기능을 제공하기 위해 Filter 인터페이스를 구현하는 자바 클래스를 생성하고, 생성된 자바 클래스를 web.xml 파일에 등록한다.

필터	기능
Request 필터	인증(사용자 인증) 요청 정보를 로그 파일로 작성 암호화 인코딩 작업
Response 필터	응답 결과 데이터 압축 응답 결과에 내용 추가/수정 총 서비스 시간 측정

필터(Filter)

필터(Filter)



필터(Filter)

Filter 인터페이스의 구현 클래스

- javax.servlet.Filter를 임포트 한다.

```
import    javax.servlet.Filter

public class 클래스 이름 implements Filter{

}
```

필터	기능
init()	필터 인스턴스의 초기화 메서드
doFilter()	필터 기능을 작성하는 메서드
destory()	필터 인스턴스의 종료 전에 호출되는 메서드

필터(Filter)

Filter 인터페이스의 구현 클래스

- init() 메서드 : FilterConfig 인터페이스 메소드의 종류

```
public void init(FilterConfig filterConfig) throws ServletException
```

메서드	설명
getfilterName()	web.xml 파일에 설정된 필터 이름 반환
getInitParameter(String name)	web.xml 파일에 설정된 매개 변수에 대한 매개변수 값을 반환

- doFilter() 메서드 : ServletRequest, ServletResponse, FilterChain 클래스 사용

```
public void doFilter(ServletRequest request,  ServletResponse response,
        FilterChain chain) throws java.io.IOException. ServletException
```

- destory() 메서드 : 필터 인스턴스를 종료하기 전에 호출하는 메서드

필터(Filter)

web.xml 파일의 필터 구성

<filter> 요소

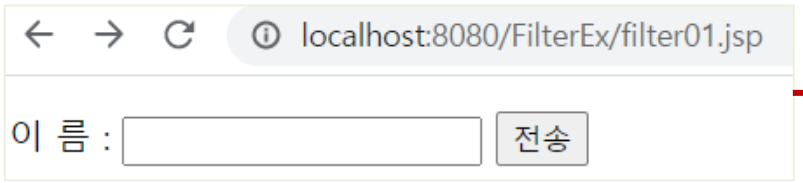
```
<filter>
  <filter-name>필터 이름</filter-name>
  <filter-class>클래스 이름</filter-class>
  <init-param>
      <param-name>매개변수 이름</param-name>
      <param-value>매개변수 값 </param-value>
  </init-param>
</filter>
```

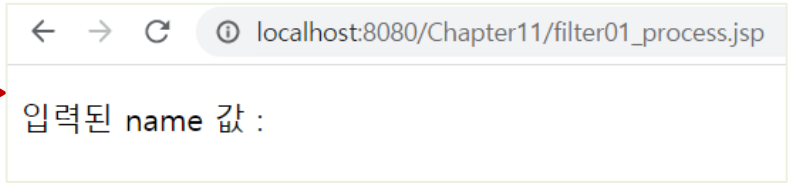
<filter-mapping> 요소

```
<filter-mapping>
  <filter-name>필터 이름</filter-name>
    <url-pattern>요청 URL 패턴</url-pattern>
  </filter-mapping>
```

필터(Filter)

폼 페이지에서 전송된 요청 파라미터를 필터로 처리하기





filter01.jsp

```
<form action="filter01_process.jsp" method="post" >
    <p>이 름 : <input type="text" name="name">
       <input type="submit" value="전송">
</form>
```

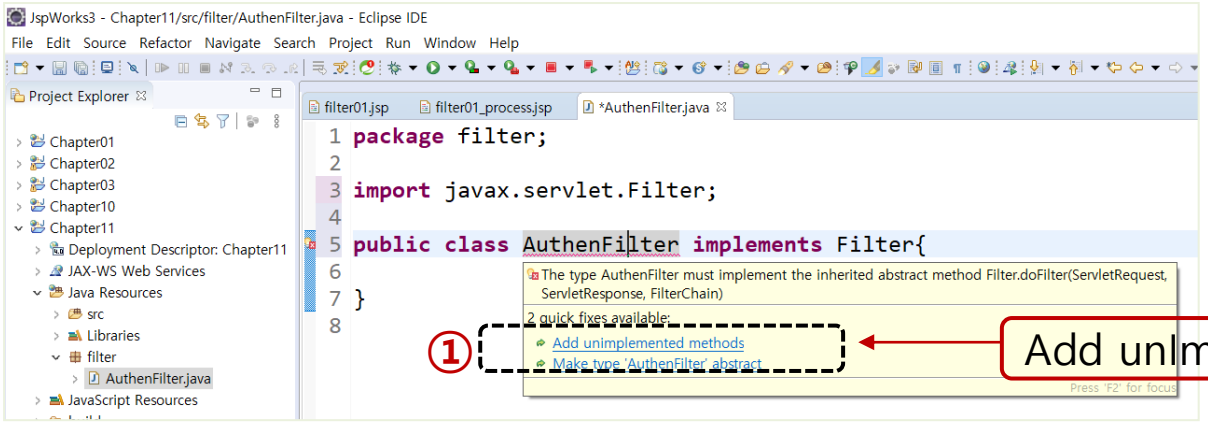
filter01_process.jsp

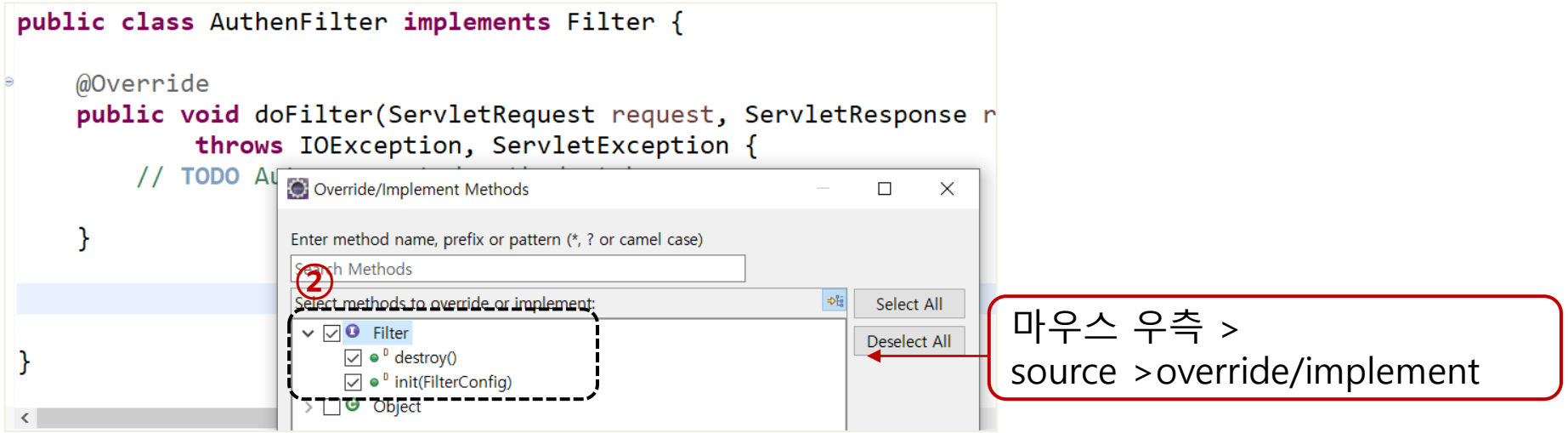
```
<%
    request.setCharacterEncoding("utf-8");

    String name= request.getParameter("name");
%>
<p>입력된 name 값 : <%=name %>
```

필터(Filter)

필터 만들기 - AuthenFilter.java





필터(Filter)

필터 만들기 - AuthenFilter.java

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;

public class AuthenFilter implements Filter{

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        System.out.println("Filter01 초기화...");
    }
```

필터(Filter)

```java
@Override
public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
    request.setCharacterEncoding("utf-8");       //한글 인코딩
    response.setContentType("text/html; charset=utf-8"); //컨텐츠 유형
    PrintWriter writer = response.getWriter();

    String name = request.getParameter("name");
    String message = "";

    if(name == null || name.equals("")) {
        message = "입력된 name 값은 null입니다.";
        writer.println(message);
        return;
    }

    chain.doFilter(request, response);  //필터 실행
}

@Override
public void destroy() {
    System.out.println("Filter02 해제...");
}
```

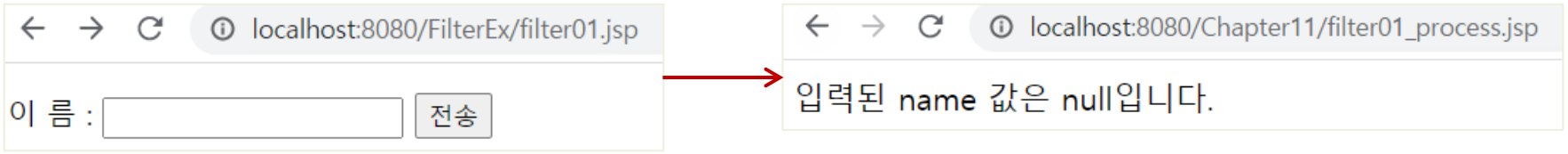
AuthenFilter.java

```
11월 11, 2020 7:03:23 오후 org.apache.cata
INFO: 서버가 [1099] 밀리초 내에 시작되었습니다.
Filter01.jsp 수행...
```

필터(Filter)

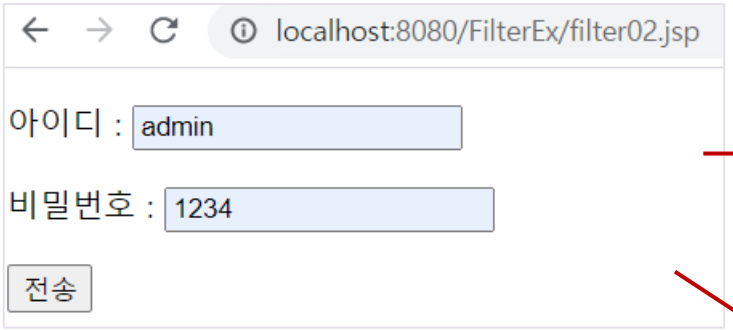
web.xml 에 등록 후 결과

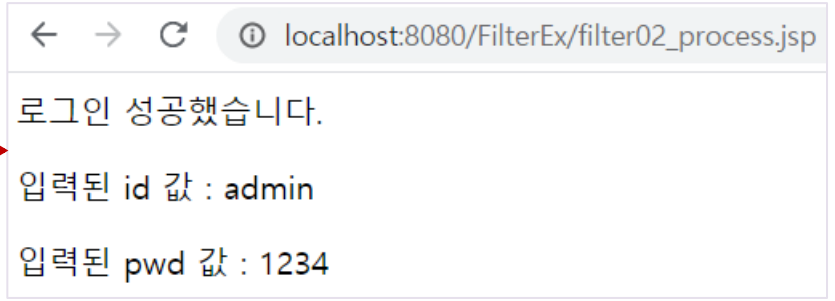
```
<filter>
  <filter-name>Filter01</filter-name>
  <filter-class>filter.AuthenFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>Filter01</filter-name>
  <url-pattern>/filter01_process.jsp</url-pattern>
</filter-mapping>
```

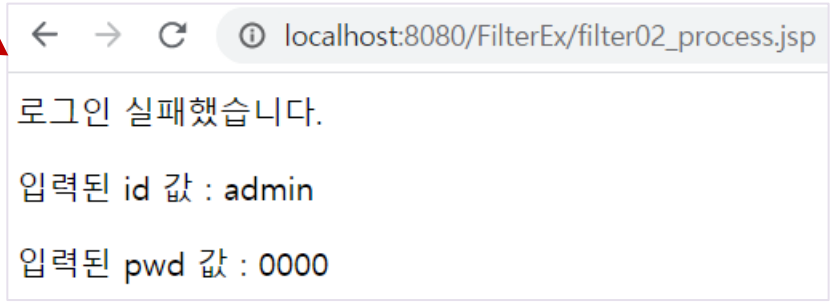


필터(Filter)

필터 처리로 매개변수와 값을 전달받아 로그인 인증 처리하기







filter02_process.jsp

```
<%
    request.setCharacterEncoding("utf-8");

    String id = request.getParameter("id");
    String pw = request.getParameter("passwd");
%>
<p>입력된 id 값 : <%=id %>
<p>입력된 passwd 값 : <%=pw %>
```

필터(Filter)

필터 처리로 매개변수와 값을 전달받아 로그인 인증 처리하기

```
public class InitParamFilter implements Filter{
    private FilterConfig filterConfig = null;  //필터 설정 객체 선언

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        this.filterConfig = filterConfig;
        System.out.println("Filter02 초기화...");
    }

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

        request.setCharacterEncoding("utf-8");       //한글 인코딩
        response.setContentType("text/html; charset=utf-8");
        PrintWriter writer = response.getWriter();
```

필터(Filter)

InitParamFilter.java

```
        String id = request.getParameter("id");
        String pwd = request.getParameter("passwd");

        String param1 = filterConfig.getInitParameter("param1");
        String param2 = filterConfig.getInitParameter("param2");

        String message = "";

        if(id.equals(param1) && pwd.equals(param2))
            message = "로그인 성공했습니다.";
        else
            message = "로그인 실패했습니다.";

        writer.println(message);    //메시지 출력

        chain.doFilter(request, response);  //필터 실행
    }

    @Override
    public void destroy() {
        System.out.println("Filter02 해제...");
    }
```

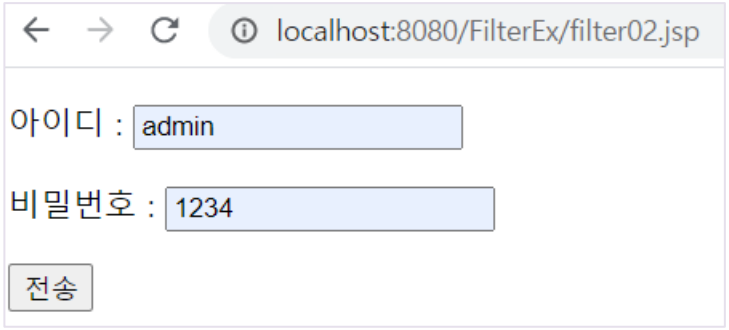
필터(Filter)

필터 처리로 매개변수와 값을 전달받아 로그인 인증 처리하기

```
<!-- 로그인 인증 -->
<filter>
  <filter-name>Filter02</filter-name>
  <filter-class>filter.InitParamFilter</filter-class>
  <init-param>
      <param-name>param1</param-name>
      <param-value>admin</param-value>
  </init-param>
  <init-param>
      <param-name>param2</param-name>
      <param-value>1234</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>Filter02</filter-name>
  <url-pattern>/filter02_process.jsp</url-pattern>
</filter-mapping>
```

필터(Filter)

로그인 인증 필터를 이용하여 파일에 로그 기록하기



monitor.log

클라이언트 IP 주소 : 0:0:0:0:0:0:0:1
현재일시 : 2021/03/26 07:26:00

필터(Filter)

로그인 인증을 이용하여 필터로 로그 기록하기

LogFileFilter.java

```
public class LogFileFilter implements Filter{

    PrintWriter writer;

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        String filename = filterConfig.getInitParameter("filename");
        if(filename==null)
            throw new ServletException("로그 파일의 이름을 찾을 수 없습니다.");
        try {
            writer = new PrintWriter(new FileWriter(filename, true), true);
        } catch (IOException e) {
            throw new ServletException("로그 파일을 열 수 없습니다.");
        }
    }
```

파일을 생성한 후 로그파일에 기록하기 위한 writer 객체 생성

예외를 강제로 발생시킴

필터(Filter)

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

    String clientAddr = request.getRemoteAddr();
    writer.printf("클라이언트 IP 주소 : %s %n", clientAddr);

    writer.printf("현재일시 : %s %n", getCurrentTime());  //getCurrentTime() 호출

    chain.doFilter(request, response);  //필터 실행
    writer.println("---------------------------------------");
}

@Override
public void destroy() {
    writer.close();
}
//현재 날짜와 시간 출력 메서드
private String getCurrentTime() { //문자열로 출력 -> DateFormat
    DateFormat formatter = new SimpleDateFormat("yyyy/MM/dd hh:mm:ss");
    Calendar calendar = Calendar.getInstance();
    calendar.setTimeInMillis(System.currentTimeMillis());
    return formatter.format(calendar.getTime());
}
```

LogFileFilter.java

필터(Filter)

로그인 인증 이용하여 필터로 로그 기록하기

```
<!-- 로그인 인증 필터 처리를 로그(log)로 기록하기 -->
<filter>
  <filter-name>Filter02_2</filter-name>
  <filter-class>filter.LogFileFilter</filter-class>
  <init-param>
      <param-name>filename</param-name>
      <param-value>c:\\logs\\monitor.log</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>Filter02_2</filter-name>
  <url-pattern>/filter02_process.jsp</url-pattern>
</filter-mapping>
```

C드라이브에 log폴더 만들기. monitor.log파일은 필터 수행되면 자동 생성됨.

필터(Filter)

Class FileWriter

java.lang.Object
    java.io.Writer
        java.io.OutputStreamWriter
            java.io.FileWriter

All Implemented Interfaces:

Closeable, Flushable, Appendable, AutoCloseable

```
public class FileWriter
extends OutputStreamWriter
```

FileWriter

```
public FileWriter(File file)
           throws IOException
```

Constructs a FileWriter given the File to write, using the platform's default charset

Parameters:

file - the File to write.