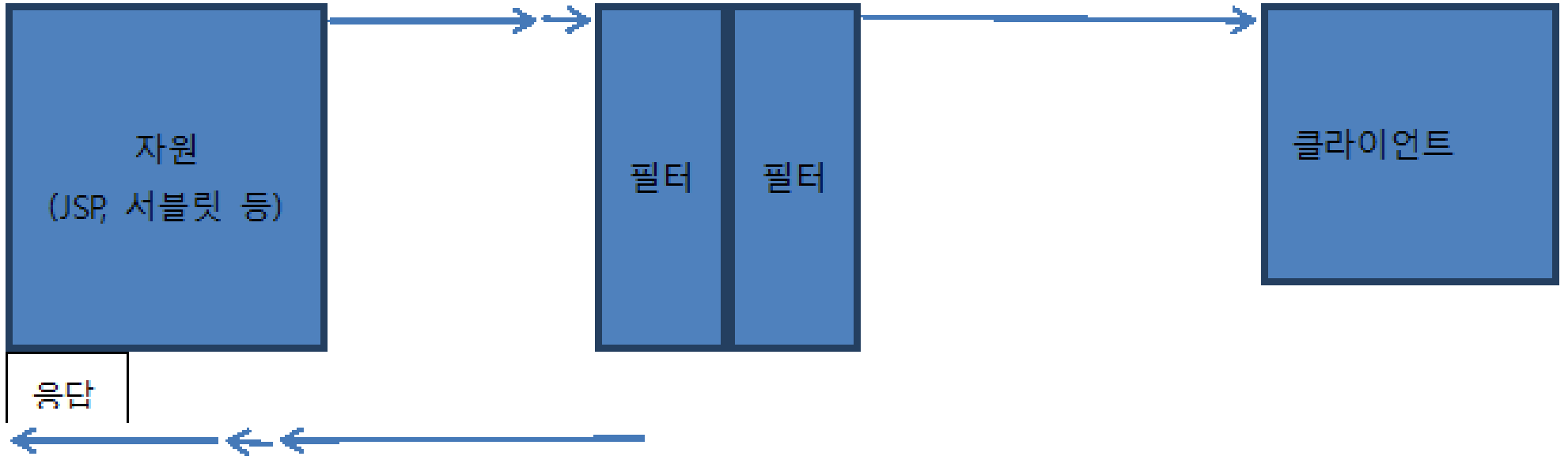
Chapter11 필터와 래퍼

뇌를 자극하는 JSP&Servlet

목차

- 필터란?
- 필터 예제
- Dispatcher
- Filter chain
- Wrapper class 란?
- Wrapper class 예제

필터란?



웹 브라우저와 웹 컴포넌트 사이에 위치해 여과기 역할을 하는 프로그램

말 그대로 클라이언트의 요청을 선 처리 하거나 서버의 자원을 가공하여 보내주는 역할

Ex) UTF-8 필터

```
/**
 * @see Filter#doFilter(ServletRequest, ServletResponse, FilterChain)
 */
public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {
	// TODO Auto-generated method stub
	// place your code here
	// 먼저 한글 처리를 하고 뒤로 보낸다.
	request.setCharacterEncoding("UTF-8"); // 리퀘스트를 utf-8로 인코딩한다.
	response.setContentType("text/html;charset=utf-8");
	System.out.println("한글 처리함.");
	// pass the request along the filter chain
	// 만약 인코딩 말고 다른 처리도 필요하다면 filter chain이라는 기법으로 여러 필터를 연결하여 필터링 할 수 있다.
	chain.doFilter(request, response);
}
```

필터 클래스를 작성한 후 과정

- 1) 필터 클래스을 컴파일
- 2) 필터 클래스의 컴파일 결과물을 웹 컨테이너의 설치
- 3) 필터 클래스를 web.xml 파일에 등록

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
    id="WebApp_ID" version="3.1">
    <display-name>filterPractice</display-name>
    <filter>
        <filter-name>encoding</filter-name>
        <filter-class>com.filterPractice.filter.EncodingFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>encoding</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

</web-app>
```

<filter-name>: 말 그대로 필터의 이름이다. <filter></filter>, <filter-mapping></filter-mapping>에 들어가는 이름이 같아야 한다.

<filter-class>: 필터가 있는 위치이다. 여기서는 com.filterPractice.fillter라는 자바 패키지에 EncodingFilter.java를 만들었다는 의미이다.
<url-pattern>: 어떤 형태의 url을 잡아낼 것인지 결정하는 부분이다.

<dispatcher> 엘리먼트

- 웹 컴포넌트 호출하는 방법 4가지

-web browser

-forward method

- Include method

- Exception auto

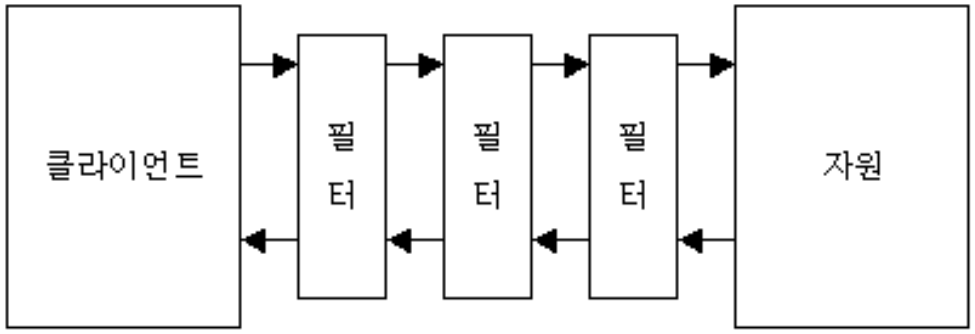
Dispatcher 엘리먼트를 통해 상황에 맞게 선택적으로 적용 가능

```
<filter-mapping>

        <filter-name>log-filter</filter-name>
        <url-pattern>/sub2/*</url-pattern>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>INCLUDE</dispatcher>

</filter-mapping>
```

Filter chain



여러 개의 필터가 모여서 하나의 체인을 형성할 때 첫번째 필터가 변경하는 요청 정보는 클라이언트의 요청 정보가 되지만, 체인의 두번째 필터가 변경하는 요청 정보는 첫번째 필터를 통해서 변경된 요청 정보가 됨

*순서가 정해지는 방법

url-pattern 엘리먼트와 servlet-name 엘리먼트가 섞여 있을때 ->

url-pattern 먼저 구성 후 servlet-name 엘리먼트 필터가 그 후 순서대로

Wrapper class 란?

1. 자료형의 효율적인 관리와

2. 자료형 은닉화를 위해 만들어진, 자료형 대체 클래스

*** Java 클래스 라이브러리의 wrapper class**

기본형 타입	wrapper class
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Charater
boolean	Boolean
void	Void

Wrapper class 예제

기본형인 int 타입을 wrapper class의 인자로 넣어 객체를 만들면 활용할 수 있는 방법이 굉장히 증가

```
Integer mInteger = new Integer(8);
mInteger.
        toString()
        equals(Object obj)
        byteValue()
        compareTo(Integer anotherInteger)
        doubleValue()
        floatValue()
        hashCode()
        intValue()
        longValue()
        shortValue()
```

Wrapper class 들이 가지고 있는 .
Parse로 시작하는 메소드 주목
-> 반환을 객체형이 아니라 기본형으로 해줌

```
String mNumString = new String("8");
int mNumber = Integer.parseInt(mNumString);
mNumber.
        cast                    ((SomeType) expr)
```