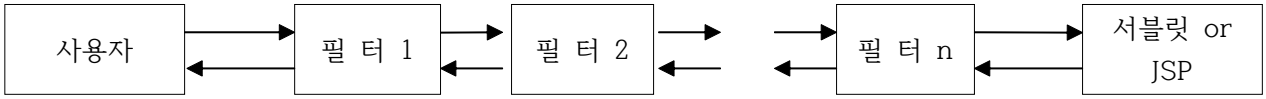
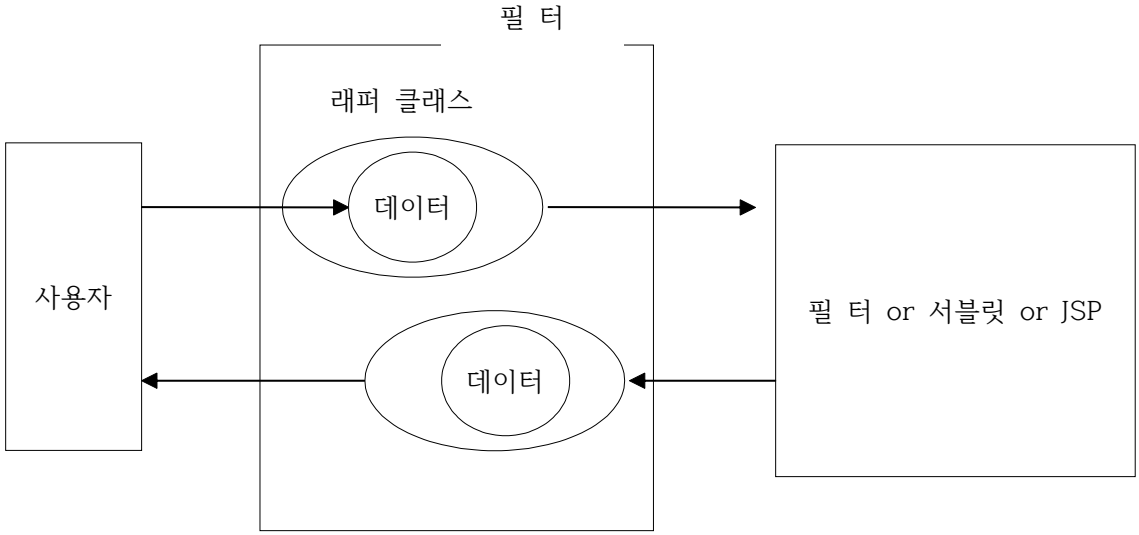
래퍼(Wrapper) 클래스

● 사용자의 데이터 변경하기

- 사용자의 요청은 필터가 대신 받아서 처리를 한 후 서블릿이나 JSP 혹은 또다른 필터로 요청을 넘겨준다.



- 필터는 사용자의 요청을 대신 받을 뿐 사용자의 요청 데이터를 변경하는 일은 하지 못한다.
- 사용자의 요청 데이터 내용을 변경할 수 있는 능력을 래퍼(Wrapper) 클래스는 가지고 있다.
- 일반적으로, 필터로 사용자의 요청을 가로챈 후 사용자의 데이터를 변경해서 서블릿이나 JSP로 넘긴다.



- 래퍼 클래스를 통과한 데이터는 즉, 변경된 데이터는 다른 필터나 서블릿, JSP로 전달되며 변경된 데이터를 전달받은 필터나 서블릿, JSP는 자신이 받은 데이터가 변경되었는지의 여부는 알 수 없으며 사용자로부터 전달된 데이터인 것으로 인식한다.

● 래퍼 클래스 작성법

- 리스너나 필터 클래스를 만드는 방식과 유사하게 HttpServletRequestWrapper이나 HttpServletResponseWrapper 클래스를 상속해서 클래스를 작성한다.
- HttpServletRequestWrapper이나 HttpServletResponseWrapper 클래스는 매개변수가 없는 기본 생성자가 없기 때문에 상속하는 클래스의 생성자에서 부모 클래스의 생성자를 작성하도록 한다.
- 사용자의 데이터를 변경하기 위한 래퍼 클래스 작성 예

```
public class MyRequestWrapper extends HttpServletRequestWrapper {
        private HttpServletRequest request;
        public MyRequestWrapper(HttpServletRequest request) {
                super(request);//기본 생성자가 없으므로, 반드시 작성해야 한다.
                this.request = request;
        }
}
```

- 사용자의 데이터를 변경하는 작업은 사용자의 데이터를 가져오는 메서드를 오버라이딩해서 변경된 데이터를 가져가도록 한다. 예를 들면, 사용자가 전송한 "파라미터"의 값을 읽어오는 getParameter()를 오버라이딩해서 변경된 데이터를 읽어오도록 한다.

- getParameter() 메서드를 오버라이딩한 예

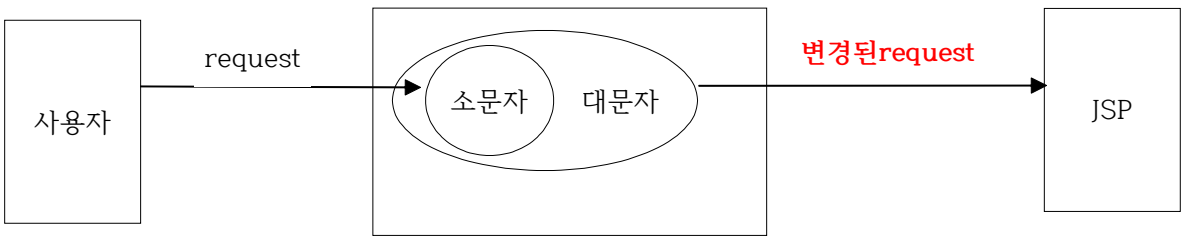
```
public class MyRequestWrapper extends HttpServletRequestWrapper {
        private HttpServletRequest request;
        public MyRequestWrapper(HttpServletRequest request) {
                super(request);//기본 생성자가 없으므로, 반드시 작성해야 한다.
                this.request = request;
        }
        public String getParameter(String name) {
                //변경된 데이터를 리턴한다.
                return 변경된 데이터;
        }
}
```

- 위에서 만든 래퍼를 필터에서 사용한다. 즉, 필터에 의해서 요청을 가로챈 뒤 사용자의 데이터를 변경해서 변경된 데이터를 리턴한다.

```
public class MyFilter extends Filter {
                ...
        public void dofilter(ServletRequest request, ServletResponse response, FilterChain chain)... {
                MyRequestWrapper requestWrapper =
                        new MyRequestWrapper((HttpServletRequest) request);
                chain.doFilter(requestWrapper, response);
        }
}
```

- 정상적인 request를 전달하는 것이 아니라, 변경된 requestWrapper를 전달한다.

● 사용자가 입력한 파라미터의 값을 모두 대문자로 변경하는 예



필 터

- 필요한 작업 : 필터작성, 소문자를 대문자로 바꾸는 래퍼작성, JSP 작성

- 소문자를 대문자로 바꾸는 래퍼 클래스

```
public class ParamUpperCaseRequestWrapper extends HttpServletRequestWrapper {
        HttpServletRequest request;
        public ParamupperCaseRequestWrapper(HttpServletRequest request) {
                super(request);
                this.request = request;
        }
        public String getParameter(String name) {//가짜 getparameter()메서드 정의
                String str = request.getParameter(name);//진짜 getParameter()메서드로 파라미터를 읽음
                if(str == null) return null;
                return str.toUpperCase();//읽은 파라미터 값을 대문자로 바꾸어서 리턴한다.
        }
}
```

- 위에서 작성한 래퍼 클래스를 사용하는 필터

```
public class ParamUpperCaseFilter implements Filter {
        public void init(FilterConfig config) throws ServletException { }

        public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
                        throws IOException, ServletException {
                ParamUpperCaseRequestWrapper requestWrapper =
                        new ParamUpperCaseRequestWrapper((HttpServletRequest) request);
                chain.doFilter(requestWrapper, response);
        }

        public void destroy() { }
}
```

- 변경된 requestWrapper를 정상적인 request인 것처럼 착각해서 작업하는 JSP

```
<%@ page contentType="text/html; charset=euc-kr" %>
<% String name = request.getParameter("NAME"); %> ← request를 사용하지만 사실은 requestWrapper이다.
<HTML>
        <HEAD><TITLE>환영 인사</TITLE></HEAD>
        <BODY>
                안녕하세요, <%= name %>님.
        </BODY>
</HTML>
```