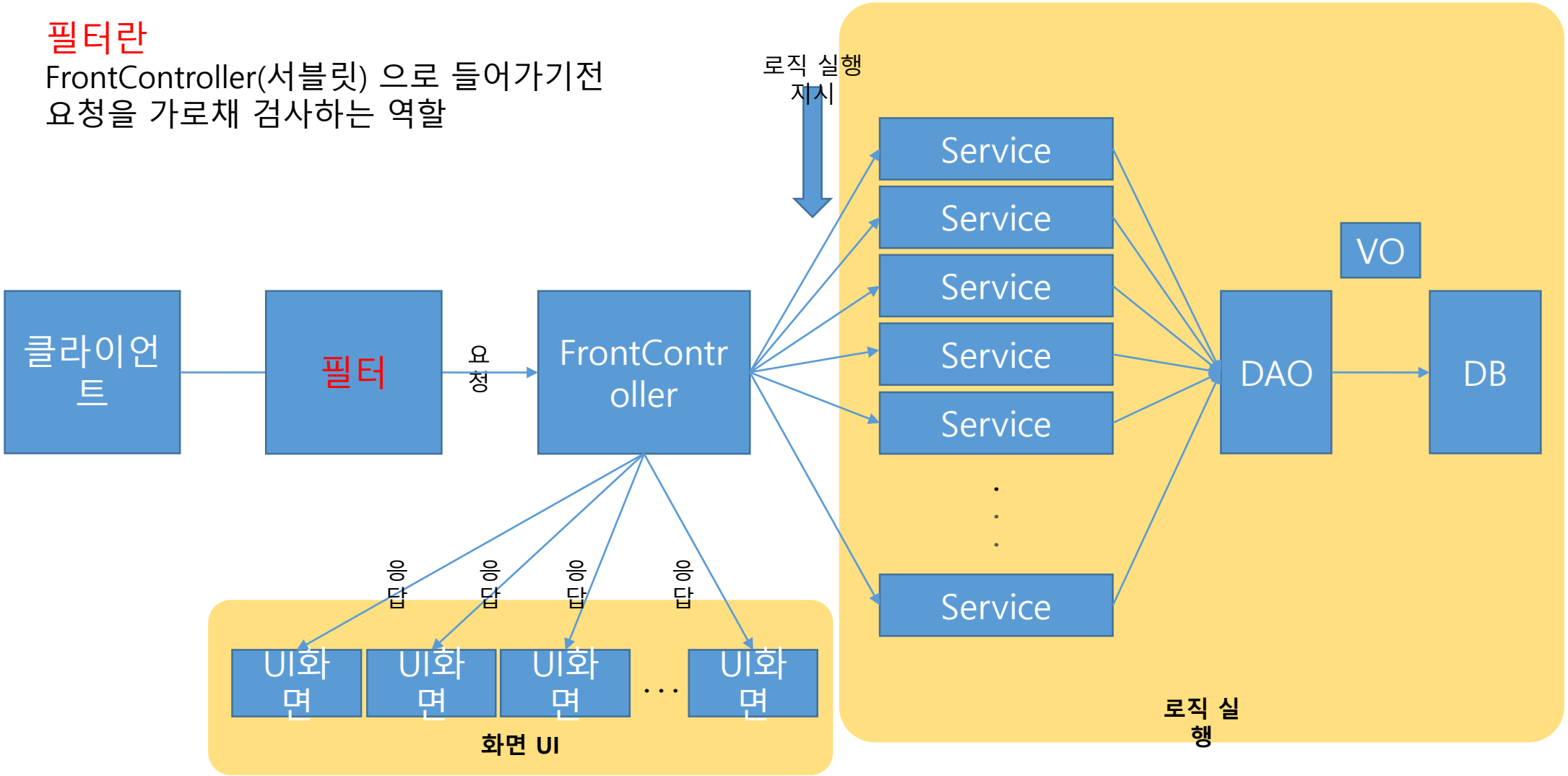
JSP

Filter란?

MVC2 전체적인 컴포넌트 설계

필터란
FrontController(서블릿) 으로 들어가기전
요청을 가로채 검사하는 역할



필터의 생성방법

1. 필터 클래스는 일반 자바 파일로 생성합니다.
2. Filter 인터페이스를 상속 받습니다
3. 일반적으로 doFilter메서드를 오버라이딩 합니다
4. 사용 후 doFilter(request, response); 메서드를 반드시 사용한다

```
public class AuthFilter implements Filter {

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

            //컨트롤러(서블릿)  으로 들어가기 전에!!!!!!
            //가로채 처리할 작업을 선언한다!!!!!!!


            //사용을 마치면 반드시 사용
            chain.doFilter(request, response);
}
```

필터의 사용방법

1. 어노테이션 방법

Ex) */board

```
//@WebFilter("*.board") //.board로 끝나는 모든요청
@WebFilter({"/board/writer.board", "/board/modify.board", "/board/update.board", "/board/delete.board" }) //특정 요청
public class AuthFilter implements Filter {

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

	}
}
```

필터의 사용방법

1. Web.xml에 선언

웹 프로젝트 설정 파일

```
<filter>
      <filter-name>FilterName</filter-name>
      <filter-class>kr.co.kim.filter.AuthFilter</filter-class>
</filter>

<filter-mapping>
      <filter-name>FilterName</filter-name>
      <url-pattern>/board/writer.board</url-pattern>
      <url-pattern>/board/modify.board</url-pattern>
      <url-pattern>/board/update.board</url-pattern>
      <url-pattern>/board/delete.board</url-pattern>
</filter-mapping>
```

필터의 예시

```
@WebFilter("/board/write.board")
public class AuthFilter implements Filter {

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

		//HttpServletRequest는 ServletRequest인터페이스의 자식 인터페이스 입니다 (형변환가능)
		HttpServletRequest req = (HttpServletRequest)request;
		HttpServletResponse res = (HttpServletResponse)response;

		//세션을 구합니다
		HttpSession session = req.getSession();
		String id = (String)session.getAttribute("user_id");

		if(id == null) {
			res.sendRedirect("board_auth_fail.jsp");
			return; //반드시 종료해야 정상흐름으로 흘러갑니다.
		}

		//세번째 매개변수인 FilterChain 클래스의 객체인 chain을 이용해서 다른 필터나 서블릿과 연결하는 코드를 반드시 작성해야 한다.
		chain.doFilter(request, response);

	}
}
```

필터의 예시

```
@WebFilter({"/board/modify.board",
            "/board/update.board",
            "/board/delete.board"
               })
public class AuthFilter implements Filter {

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

            //HttpServletRequest는 ServletRequest인터페이스의 자식 인터페이스 입니다 (형변환가능)
            HttpServletRequest req = (HttpServletRequest)request;
            HttpServletResponse res = (HttpServletResponse)response;

            //세션을 구합니다
            HttpSession session = req.getSession();
            String id = (String)session.getAttribute("user_id");

            if(!id.equals(request.getParameter("writer")) ) {
                  res.sendRedirect("board_auth_fail.jsp");
                  return; //반드시 종료해야 정상흐름으로 흘러갑니다.
            }

            //세번째 매개변수인 FilterChain 클래스의 객체인 chain을 이용해서 다른 필터나 서블릿과 연결하는 코드를 반드시 작성해야 한다.
            chain.doFilter(request, response);

      }
}
```

필터를 여러 개 등록하는 방법

- 필터를 여러개 등록하는 방법은 web.xml에 <filter>태그를 여러번 사용해 각각의 필터를 등록하면 됩니다.

필터 순서 지정

- 서버 시작시 web.xml 설정을 위에서 아래로 읽어들일때 <filer-mapping>이 정의된 순서를 기준으로 필터체인의 정렬 순서를 정의합니다.
- 따라서 다음과 같이 설정하는 경우 firstFilter -> secondFilter 순서로 필터체인이 형성됩니다.

```
<filter>
	<filter-name>firstFilter</filter-name>
	<filter-class>test.FirstFilter</filter-class>
</filter>
<filter>
	<filter-name>secondFilter</filter-name>
	<filter-class>test.SecondFilter</filter-class>
</filter>

<filer-mapping>
	<filter-name>firstFilter</filter-name>
	<url-pattern>맵핑경로</url-pattern>
</filer-mapping>

<filer-mapping>
	<filter-name>secondFilter</filter-name>
	<url-pattern>맵핑경로</url-pattern>
</filer-mapping>
```