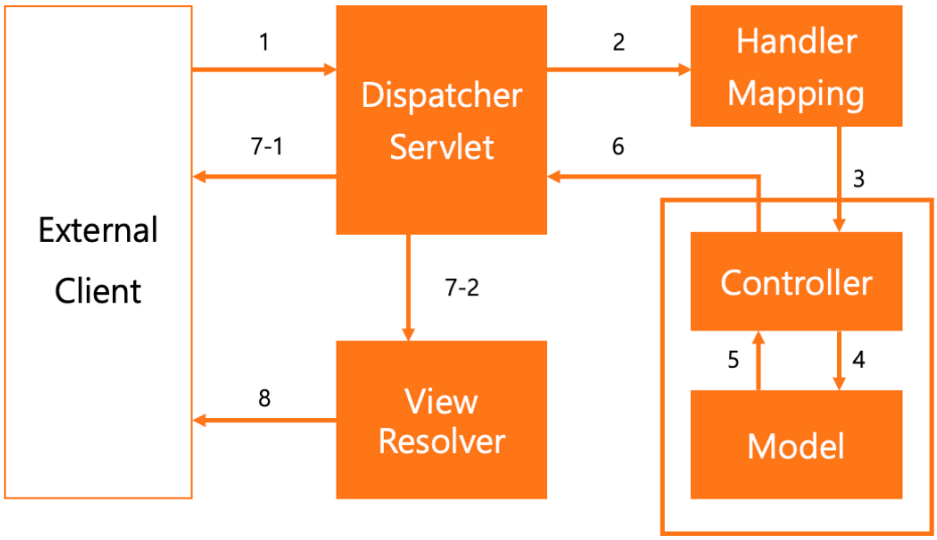
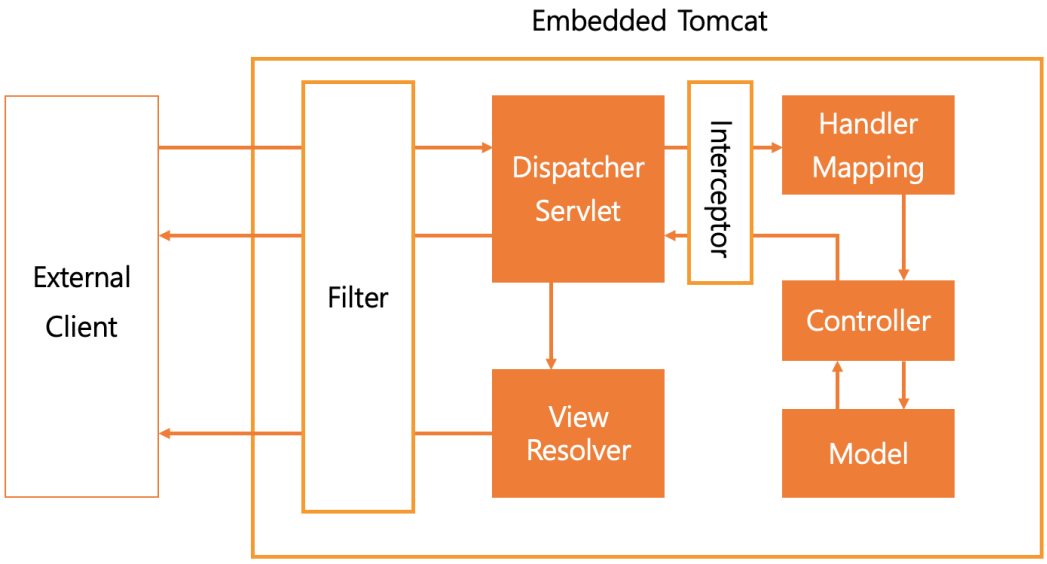
Chapter 7-2 Filters & Interceptors



서버가 받게되는 모든 요청 / 응답에 기능을 적용하고 싶을 때는?

모든 사용자가 똑같이 들어가야 할 기능

로그인 기능 ..



Filter는 Spring 외부에서, Interceptor는 Spring 내부에서

Tomcat이 둘러싸고 있음

조작 가능한 곳 :

Filter(외부와 서브렛 사이에서) - 자바로 만든 웹서버에서 사용하게 되는 인터페이스

빈으로 등록하면 Filter로 등록하게 해줌

Handler Interceptor : handler mapping 요청 처리

자바 서버, 스프링 프레임 워크에서 제공하는 기능

→ 요청과 응답이 오고가는 와중에 기능을 적용시킬 수 있도록 만들어진 인터페이스

AOP : 함수의 이름을 런타임에서 실행중에 알아냄으로써 그에 따른 행동에 따라 바꿔주고~

Filter

```
Params: request – The request to process
        response – The response associated with the request
        chain – Provides access to the next filter in the chain for
        this filter to pass the request and response to for further
        processing
Throws: IOException – if an I/O error occurs during this filter's
        processing of the request
        ServletException – if the processing fails for any other
        reason
public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException;
```

doFilter() 함수를 구현한다.

Filter

```
Params: request – The request to process
        response – The response associated with the request
        chain – Provides access to the next filter in the chain for
        this filter to pass the request and response to for further
        processing
Throws: IOException – if an I/O error occurs during this filter's
        processing of the request
        ServletException – if the processing fails for any other
        reason
public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException;
```

FilterChain 변수를 사용해 filter의 전후를 구분한다.

Spring 밖에서 !!

Interceptor

```java
@Override
public boolean preHandle(
        HttpServletRequest request,
        HttpServletResponse response,
        Object handler) throws Exception {
    return true;
}
```

preHandle(), postHandle(), afterCompletion()으로 구현할 함수가 나눠져 있다.

1) preHandle()

- 컨트롤러가 호출되기 전에 실행됨
- 컨트롤러가 실행 이전에 처리해야 할 작업이 있는경우 혹은 요청정보를 가공하거나 추가하는경우 사용
- 실행되어야 할 '핸들러'에 대한 정보를 인자값으로 받기때문에 '서블릿 필터'에 비해 세밀하게 로직을 구성할수 있음
- 리턴값이 boolean이다. 리턴이 true 일경우 preHandle() 실행후 핸들러에 접근한다. false일경우 작업을 중단하기 때문에 컨트롤러와 남은 인터셉터가 실행되지않는다.

2) postHandle()

- 핸들러가 실행은 완료 되었지만 아직 View가 생성되기 이전에 호출된다.
- ModelAndView 타입의 정보가 인자값으로 받는다. 따라서 Controller에서 View 정보를 전달하기 위해 작업한 Model 객체의 정보를 참조하거나 조작할수 있다.
- preHandle() 에서 리턴값이 fasle인경우 실행되지않음.
- 적용중인 인터셉터가 여러개 인경우, preHandle()는 역순으로 호출된다.
- 비동기적 요청처리 시에는 처리되지않음.

3) afterCompletion()

- 모든 View에서 최종 결과를 생성하는 일을 포함한 모든 작업이 완료된 후에 실행된다.
- 요청 처리중에 사용한 리소스를 반환해주기 적당한 메서드 이다.
- preHandle() 에서 리턴값이 false인경우 실행되지 않는다.
- 적용중인 인터셉터가 여러개인경우 preHandle()는 역순으로 호출된다.

- 비동기적 요청 처리시에 호출되지않음.

Interceptor

```java
@Override
public void postHandle(
        HttpServletRequest request,
        HttpServletResponse response, Object handler,
        ModelAndView modelAndView) throws Exception {

}
```

ModelAndView 객체를 조작할 수 있다.

Filter

- Jakarta Servlet API의 일부
- doFilter() 함수를 구현
- 요청, 응답을 조작할 수 있음
- Buisness Logic과 무관한 기능 구현에 사용
  - 보안
  - 데이터 인코딩

HandlerInterceptor

- Spring Application의 일부
- preHandle(), postHandler(), afterCompletion() 함수를 구현
- Buisness Logic과 연관성이 높은 기능 구현에 사용
  - 사용자 인증
  - API 처리 내용