**Spring Web Project**

201620900 소프트웨어학과 심재철

<https://devpad.tistory.com/24> - spring boot에 과정과 구조에 대해 자세히 적혀있다.

**(1) 스프링 실행 순서**

1. 클라이언트가 Request 요청을 하면, DispatcherServlet이 web.xml에 등록된 요청만 로챈다.

2. DispatcherServlet이 가로챈 요청을 HandlerMapping에게 보내 해당 요청을 처리할 수 있는 Controller를 찾는다.

3. 실제 로직 처리 (Controller -> Service -> DAO -> DB -> DAO -> Service -> Controller)

4. 로직 처리 후 ViewResolver를 통해 view 화면을 찾는다.

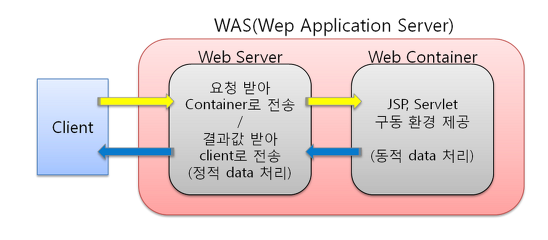
5. 찾은 view 화면을 View에 보내면 이 결과를 다시 DispatcherServlet에 보내고, DispatcherServlet는 최종 클라이언트에게 전송한다.

**사용자 요청(Request) -> Servlet Filter -> Dispatcher Servlet -> Interceptor -> Controller**

**(2) web.xml 설정 파일**

WEB vs WAS

WAS (동적 서버 콘텐츠를 수행) -> 웹 서버 + 웹 컨테이너(JSP와 Servlet을 실행시킬 수 있는 SW)



apache: http web server

tomcat: WAS (Web Application Server)

!! web.xml은 웹 application 설정 파일이다.

<servlet> : DispatcherServlet 설정을 위한 상위 태그

<servlet-name> : 변수 명

<servlet-class> : 어떤 클래스를 가지고 DispatcherServlet을 구현할

<init-param> : servlet에 대한 설정 정보

<load-on-startup> : 서블릿에 대한 우선 순위

<servlet-mapping> : 서블렛이 <url-pattern>에서 지정한 패턴으로 클라이언트 요청이 들어오면 해당 요청을 가로챈다.

**<이외의 태그>**

<welcome-file> : 요청 url이 파일을 가르키지 않고 디렉토리까지만 있을 때 해당 디렉토리 중 보여줄 jsp을 설정한다.

<filter> : init(), destroy(), doFilter()가 doFilter를 통해 클라이언트 요청 때마다있는ServletRequest, ServletResponse를 가지고 요청과 응답을 조작할 수 있습니다. 그리고 FilterChain을 통해 조작 이후 요청을 원래 목적지인 서블릿으로 전달한다.

<listener> : 이벤트 리스너

**(3) servlet-context.xml 설정 파일**

**!! Servlet에 대한 설정 파일이다.**

<beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<beans:property name="prefix" value="/WEB-INF/views/" />

<beans:property name="suffix" value=".jsp" />

</beans:bean>

: prefix, suffix 설정을 통해 일일이 전체경로와 .jsp를 붙이지 않아도 되도록 도와준다.

(controller 가 model을 return하면 dispatcher servlet이 파일을 찾을 때 쓰이는 정보)

<context:component-scan base-package="com.company.devpad" />

: 스프링에서 사용하는 bean을 일일이 xml에 선언하지 않고도 필요한 것을 어노테이션(Annotation)을 자동으로 인식

<annotation-driven />

: @Controller 어노테이션을 감지하여 해당 클래스를 Controller로 등록할 수 있도록 해주는 태그

<resources mapping="/resources/\*\*" location="/resources/" />

: 정적인 html문서 같은 웹 리소스들의 정보를 기술하는 태그

**(4) 자바 컨트롤러 파일(Java Controller File)**

@Controller

public class HomeController {

 @RequestMapping(value = "/", method = RequestMethod.GET)

public String home(Locale locale, Model model) {

   return "home";

}

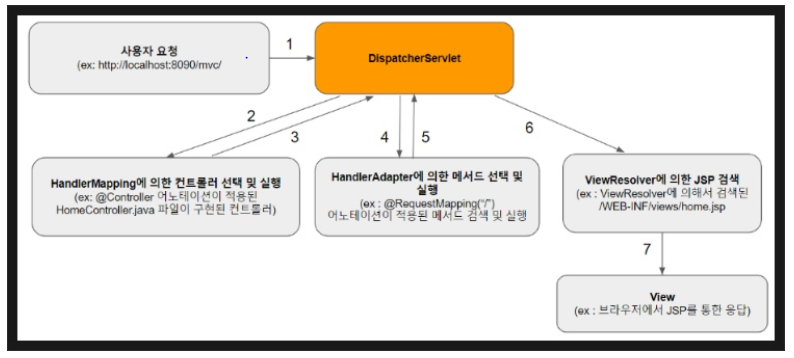
:hadlermapping에 의해 컨트롤러가 결정되고 hadler adapter에 의해 실행 method가 request mapping 정보를 가지고 결정된다. (get방식으로 요청이 들어올 때)

**(5) JSP 파일**

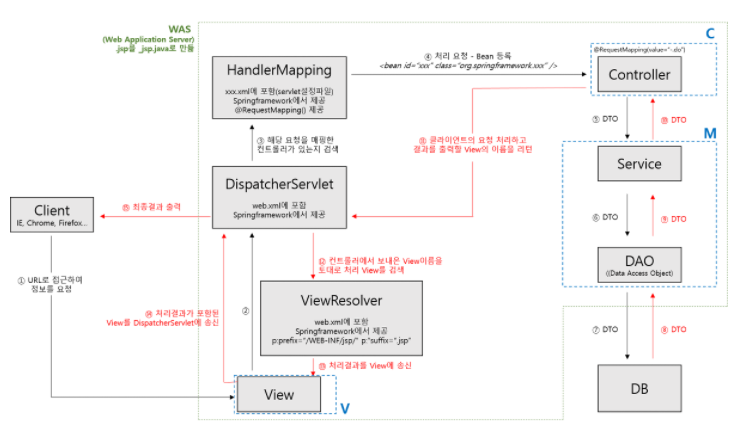
JSP 파일은 Model 객체를 넘겨받고 그 Model 객체 안의 속성 값들의 정보를 ${} 기호에 표현된 부분에 치환한다.

**(6) root-context.xml**

 Service, Repository(DAO), DB등 비즈니스 로직과 관련된 설정



**(6) 전체적인 과정**



Spring Security 인증, 권한에 대한 설정을 할 수 있다. (interceptor 이용해서 가능)

Interceptor vs servlet filter

현재 프로젝트 구조 파악

사용자 url 입력으로 요청 (login이 아닐 때)

* webconfig에서 해당 url에 대한 권한을 가지고 있는 지 보고 interceptor가 추가 되있으면 interceptrorf를 이용하게 된다.
* interceptor에서는 url이 mapping되는 controller로 넘기기 전에 token에 해당하는 정보가 login 정보를 가지고 있는지 있다면 권한이 있는지 확인한다.
* 없다면 login페이지로 넘기는 작업을 한다.
* interceptor의 일이 끝나면 controller가 url에 대한 요청을 받는다.
* interceptor에 대한 생성과 url mapping은 WebMvcConfigurer를 확장하는 WebMVConfig에서 구현하고 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그렇다면 login할 때 request에서 token을 가져올텐데, 이 request에 token이 들어가는 시점이 어떻게 되나?

회원가입 및 만료시 토큰을 생성해준다.

httpsender 함수에서 token을 헤더에 추가시켜 전달하는 과정이 있다.

서버 기반 인증 – statefull , 서버에서만 session을 가지고 있다.

토큰 기반 인증 – stateless . 클라이언트가 tokend을 가지고 있다.

대칭키 VS 공개키

대칭키: 같은 키로 암호화와 복화화가 가능

공개키: 키를 두 개 가지고 있으며, 하나로 암호화, 다른 하나로 복호화가

가능하며 또 다시 다른 하나로 암호화, 하나로 복호화가 가능.

=> 데이터를 보존하는 차원이 아닌, 클라이언트가 서버가 신뢰성이 있는지

알아볼 수 있게 된다. (SSL)

CA: SSL 인증서를 제공하는 기업들

-참고: 생활코딩