Session-2

WAS가 브라우저의 요청을 받으면, HttpServletRequest, HttpServletResponse 객체가 생성한다. 이걸 dispatch servlet에 전달한다. 컨트롤러 파라미터로 HttpServletRequest같은거 사용하면

서블릿 : HttpServlet 이란 객체를 상속받는데, HttpServlet은 Service로 request,response가 있는데

브라우저에서 서블릿으로 요청을 날리면, 모든 요청을 처리해주는 디폴트 서블릿이 있다.

정적컨텐츠를 처리하기 위해서 만들어진 디폴트 서블릿 – was 에서 기본적으로 하나 잇다.

스프링을 사용하면 모든 요청은 dispatcher servlet얘가 잇어서ㅓ default servlet같이 모든 요청을 받아서 response 주는 애가 있는데, 만약에 dispatcher 에 default servlet을 매핑시켜주면 알아서 실행이 된다~

Dispatcher servlet은 web.xml파일에 설정을 해줘야 한다. Servlet 3.0미만엔 반드시 xml파일이 있어야하는데 그 이상 버전에는 web.xml파일을 java config로도 만들 수 있다.

스프링 부트를 사용하면 dispatcher servlet을 설정할 필요가 없다. = web.xml사용할 일이 없다. 스프링부트는 오토 컨피그라는게 있는데, 스프링부트는 web.xml에 해당하는 java config 를 이미 가지고 있다.

자바에서는 front controller라고 잇는데 이게 dispatcher servlet임. 모든 요청을 받았는데, 이건 모든 요청만 받고 어떤 요청인지에 따라서 Controller에게 분배한다.

하나의 컨트롤러는 여러 개의 url을 처리할 수 있다. Ex) 상품 조회수정삭제 다 처리할 수 있는 controller.

Dispatcher servlet은 web Application context를 가지고 있따. 이놈이 bean factory인데 bean을 관리해준다.

스프링은 bean을 싱글턴으로 관리한다. Bean1개에 메모리 1번올린다. 컨트롤러 100개면 100번올리고 dispatcher servlet이 각각의 컨트롤러를 안다. 빈으로 등록할 때 관리하는 놈임.

컨트롤러는 싱글턴이지만 동시에 실행이 가능해서 문제가 생길 수 있다. 동시에 접근할 수 있다는 생각을 항상 하고..

ComponentScan

@Component가 붙은 애를 찾아서 bean으로 등록한다.

RestController

얘는 컨트롤B 하면

Controller

를 상속하고있고 얘를 컨트롤 B하면

Component

가있다. 얘가 HelloController에 붙어있으니

결국

Web1Application

얘가 HelloController를 bean으로 등록한다.

@ResponseBody : 컨트롤러 메서드의 결과를 그대로 브라우저로 출력하겠다. 얘가 없으면 뷰의 이름이 된다. 브라우저가 출력한다는 말은 응답 body란에 보낸다는 말임. get방식에는 body에 값을 담아서 보낼 수 없다. 하지만 post, put방식은 많은 양이 필요해서 body에 보낼 수 있다.(body에다가는 xml,json같은걸 담을 수 잇따. 그래서 컨트롤러에게 내가 보내는 데이터 형식이 무엇인지 알려줘야한다. 그때 사용되는 헤더 타입이 Content-Type이다. Restlet에 설정해줘ㅜ야함.

참고로 파라미터는 header에 달려 request가 간다.

@GetMapping 어노테이션 안에

RequestMapping

이게 잇는데 이건

뷰 이름이 사용된다 = 화면으로 응답하는 애플리케이션 : @Controller

JSON / XML 데이터로 응답하는 애플리케이션 @RestController -> 브라우저에 그대로 출력.

컨트롤러가 뷰 이름을 리턴하면, 사용자가 요청한 패스의 확장자라던지 헤더정보의 accept 같은 정보를 가지고 뷰 이름이 어떤 확장자로 보여주면 좋은 것인지 구분해서 view resolver가 판단해서 xml로 보여줄 수 도 있고 json도 보ㅑ여줄 수 도 있고 html로 보여 줄 수도 있다.