jcenter 와 mavencentral의 차이

테스트 코드 작성의 이점 53p

@SpringBootApplication 설명 57p

Jar 파일 +

내장 WAS 사용을 권장하는 이유 = ‘언제 어디서나 같은 환경에서 스프링 부트를 배포할 수 있기 때문’, 외장 WAS를 쓸 때 필요한 것들… , 톰캣이란? – 서블릿으로 이루어진 자바 어플리케이션(공부하기)

58p

RestController, GetMapping 어노테이션 설명 60p

테스트 코드 관련 어노테이션들 설명 62p

롬복 관련 어노테이션

@RequiredArgsConstructor : 선언된 모든 final 필드가 포함된 생성자를 생성해 줍니다.

final이 없는 필드는 생성자에 포함되지 않습니다.

72p

Junit의 기본 assertThat이 아닌 assrtj의 assertThat을 사용하는 것의 이점 (백기선 유튜브 링크 포함)

74p

@RequestParam 에 대한 설명(외부에서 API로 넘긴 파라미터를 가져오는 어노테이션)

75p

// 인텔리제이 output 창 한글이 깨질 시, shift 2번 입력해 검색창을 띄우고

// VM option 검색하여 커스텀 Edit Custom VM options 로 들어가서

// 마지막줄에 -Dfile.encoding=UTF-8 을 입력하고 인텔리제이 재시작한다

test 코드의 파라미터를 넘기는 법, json 응답값을 필드별로 검증하는 메소드 등

param은 String만 가능하므로 숫자, 날짜 등의 데이터는 문자열로 변경해야함

3장 78p  JPA로 데이터베이스 다루기

MyBatis, iBatis는 ORM이 아닌 SQL Mapper이다

JPA 소개 80p

JPA의 역할과 Spring Data JPA의 특성과 장점 82~84p

도메인이란?(공부하기) 87p

JPA 어노테이션 설명 및 도메인의 포스트 클래스 작성, 중요 어노테이션을 클래스에 가깝게 배치하는 이유 : 새 언어로 전환 시 삭제가 용이하게, 엔티티 클래스는 실제로 DB의 테이블과 매칭될 클래스다

 89p

\*\* @NoArgsConstructor 는 기본 생성자를 만들어주고, @RequiredArgsConstructor 는 final 필드만을 포함하는 기본 생성자를 만들어준다.

생성자와 빌더에 대해 93p

@After, jpa 테스트 등  ~~~.save : insert/update 쿼리 수행. id 값이 있다면 update 없으면   
insert 한다 , @SpringBootTest를 사용하면 자동으로 H2 DB를 실행해줌 97p

API 만들기 100p

Service는 비즈니스 로직을 처리하지 않는다. 단지 트랜잭션, 도메인 간 순서 보장의 역할만 한다. Spring 웹 계층에 대해 101p566.

+

컨트롤러, 서비스, 저장 api 105p

Entity 클래스가 Request/Response 클래스와 비슷해도, 이렇게 사용하면 안된다. Entity 클래스는 데이터베이스와 맞닿은 핵심 클래스임. Entity 클래스를 기준으로 테이블이 생성되고, 스키마가 변경된다. 수많은 서비스 클래스나 비즈니스 로직들이 Entity 클래스를 기준으로 동작한다. 이런 Entity 클래스가 변경되면 많은 클래스에 영향을 끼치지만, Request/Response 용 Dto는 View위한 클래스라 정말 자주 변경이 필요하다

-> View Layer 와 DB Layer의 역할 분리

@WebMvcTest 는 JPA기능이 작동하지 않으므로, JPA 기능 테스트 시에는 @SpringBootTest와 TestRestTemplate를 사용하면 된다. 110p

JPA의 영속성, 더티 체킹 113 ~ 114p

JPA Auditing 으로 생성시간/수정시간 자동화하기 121p

JPA Auditing 어노테이션을 활성화하려면 어플리케이션에 @EnableJpaAuditing을 추가해야함

4장 머스테치로 화면 구성하기(뷰 템플릿) 125p

자바스크립트 코드안에 자바코드를 작성하면 제대로 동작하기가 어렵다.

JSP와 같은 서버 템플릿 엔진은 서버에서 구동되고 자바스크립트는 브라우저에서 구동되기 때문에 서로 서로 따로 실행되며, 실행되는 도중에는 서로의 코드는 단순한 문자열로 취급하기 때문  126~127p

템플릿 엔진들의 단점 129p

머스테치의 장점 : 로직 코드를 사용할 수 없어 View의 역할과 서버의 역할을 명확하게 분리, 하나의 문법으로 클라이언트/서버 템플릿을 모두 사용 가능하다

레이아웃 방식 : 공통 영역을 별도의 파일로 분리하여 필요한 곳에서 가져다 쓰는 방식 135p

js 파일에서 바로 함수를 작성하지 않고 변수의 속성으로function을 추가한 이유 142p

144p 게시글 등록 기능을 추가할 때, index.js 파일을 불러들이는 코드를 작성하고나서 어플리케이션을 동작시키니 Uncaught SyntaxError: Unexpected token '<' 라는 에러가 발생했다. 원인을 도저히 알 수 없었다.. 자바스크립트를 JSON 으로 파싱하는 것 때문에 보통 발생하는 에러라는데 그에 필요한 해결법을 적용해도 고쳐지지 않았다.

결국 확인해보니 IndexController 에서 기본 URL인 “/” 에 대해서

@GetMapping(“/”) 을 해야하는데 (“/”)을 빼먹어서  @GetMapping 만 적어놔서 생긴 문제였다.

컨트롤러를 수정하니 제대로 동작하게 되었다.

하지만 의문점이 있다.  @GetMapping(“/posts/save”)는 제대로 작성해놓은 상태였고, 메인 페이지를 거치지 않고 바로 (“<http://localhost:8080/posts/save>)로 접근을 하는데도 왜 기본 index 주소에 매핑이 안되어있다고 에러가 발생하는 걸까? 심지어 매핑안해놔도 기본으로 index를 찾는 속성때문에 홈페이지 자체는 제대로 출력되었고, index.js 로딩만 에러가 발생했다. 정확히 어떤 이유에서 발생했는지는 나중에 제대로 알아보아야 겠다…

자동으로 index를 인식할 때는 mustache의 경로 탐색이 제대로 이루어지지 않는건지 어떤 이유인지 알아봐야겠다.

148p @Transactional(readOnly=true)  조회속도 개선하는 법

150p Model : 서버 템플릿 엔진에서 사용할 수 있는 객체를 저장할 수 있다.

@PathVariable 공부하기

163p 5장 스프링 시큐리티와 OAuth 2.0으로 로그인 기능 구현하기

172p 승인된 리디렉션 URI

174p google oauth scope에서 openid를 빼는 이유

180p~ security설정 config 파일 설명

185p SessionUser : User클래스를 쓰지 않고 새로 만들어서 쓰는지는 나중에 설명 = 188p

User 클래스에 직렬화를 구현하지 않았기 때문에 User 클래스를 세션에 저장하려하면 에러가 발생하는데, User 클래스에 직렬화 코드를 넣으면 다른 엔티티와 관계가 형성되었을 때 이슈가 발생할 확률이 높아진다. 따라서 직렬화 기능을 가진 세션 Dto를 추가로 만들어서 운영 및 유지보수 때 이점을 갖게한다.

195p 어노테이션으로 기능개선하기 - 어노테이션 생성

199p HandlerMethodArgumentResolver는 항상 WebMvcConfigure의 addArgumentResolvers()를 통해 추가해야 한다

@Component 공부하기

201p 세션 저장소를 선택하는 기준

216p @SpringBootTest 에서 MockMvc를 사용하는 방법

217p WebApplicationContext context

220p @WebMvcTest, 이 어노테이션은 일반적인 @Configuration은 스캔하지 않는다.

@Configuration 어노테이션 동작 원리 공부하기

224p 6장 AWS 서버 환경 만들기 - EC2

226p IaaS, PaaS, SaaS

258p 아마존 리눅스 1 서버 생성 시 꼭 해야 할 설정들

265p 7장 AWS에 데이터베이스 환경을 만들어보자-AWS RDS

267p 마리아 db 추천 이유

273p RDS 처음 생성 시 필수 설정들

295p 8장 EC2 서버에 프로젝트 배포하기

298p 배포스크립트

300~302p 배포스크립트 작성, 및 설명

306p 스프링 부트 프로젝트로 RDS 접근하기, MariaDB에서 스프링부트 프로젝트 실행하기 위해 필요한 작업들

\*\*

발생한 문제 : 로그인 API는 제대로 호출되는데 로그인을 해도 계속 /login 페이지로 리다이렉트 되는 문제가 발생했음. 수많은 뻘짓 끝에 서버 로그를 보고 user 테이블을 안 만들어놨다는 것을 깨달음.

원래 세션정보는 서버에 저장되는데, DB에 저장하기로 설정을 해놨기 때문에 로그인을 하면 세션정보가 db에 저장된다. 근데 user 테이블이 없으니, 로그인 해서 얻은 권한을 가질 수 없게 되어서 계속 /login 페이지로 리다이렉트 되었던 것(권한이 없는 url에 접근하면 /login으로 리다이렉트)

320p 9장 배포 자동화

CI , CD

CI : VCS 시스템에 PUSH가 되면 자동으로 테스트와 빌드가 수행되어 안정적인 배포 파일을 만드는 과정을 CI라고 한다(Continuous Integration) 이 빌드 결과를 자동으로 운영 서버에 무중단 배포까지 진행되는 과정을 CD(Continuous Deployment)라고 한다.

361p Travis CI는 S3로 특정 파일만 업로드가 안되고, 디렉토리 단위로만 업로드 가능하다. 따라서 항상 deploy 디렉토리를 생성해서 사용해야함.

365p CodeDeploy 로그 확인

368p 무중단 배포

무중단 배포방식

aws 블루 그린 무중단 배포

도커를 이용한 웹서비스 무중단 배포

nginx를 이용한 무중단 배포

L4 스위치(대형 인터넷 기업)

쉽고 저렴한 nginx 방법

1대의 nginx와 스프링 부트 jar 2대를 사용한다.

nginx는 80(http), 443(https) 포트를 할당하고

springboot1 은 8081 포트

springboot2 는 8082 포트 로 실행한다

평소에는 nginx는 하나의 스프링부트로 요청을 전달.

신규 배포가 필요할 때 연결되지 않은 스프링부트 쪽으로 배포를 한다.

배포가 완료되고 새로운 버전의 스프링부트가 정상적으로 구동중이면

nginx는 새로운 스프링부트로 reload한다

다시 배포가 필요하면 다른 사용하지 않는 스프링부트로 위 과정을 반복

쉘 스크립트 sh파일에서 첫 줄에

#!/usr/bin/env bash 의 의미가 뭐지

엔진엑스는 중요한 설정을 반영하려면 재시작해야 하지만 리로드를 사용하면 끊김 없이 사용이 가능하다

그러므로 잦은 변경이 필요한 설정의 경우

외부 설정파일로 사용하여 리로드로 반영되도록 설계하자

명령어

nginx 설정 파일 오픈

sudo vim /etc/nginx/nginx.conf

399p 11장 1인 개발 시 도움이 될 도구와 조언들