Sec02

Controller 에서 @ 사용하기

작성자: 강태경



# Exam04

@ModelAttribute 객체형태로 변환하는 어노테이션

* @ModelAttribute : QueryString(쿼리스트링) 방식으로 전송되는 매개변수를 객체의 속성을 통해 전달받을 수 있음  
  QueryString 매개변수명 == 객체의 속성명 (일치)
* Lombok 라이브러리 : 특수한 기능을 제공하는 어노테이션이 많음  
  @Setter : 모델 클래스의 위에 사용해서 Setter 함수 자동생성  
  @Getter : 모델 클래스의 위에 사용해서 Getter 함수 자동생성  
  @ToString: 모델 클래스의 위에 사용해서 ToString() 함수 자동 재정의  
  @NoArgsConstructor : 모델 클래스에 사용, 기본 생성자 자동 생성  
  @AllArgsConstructor : 모델 클래스에 사용, 속성에 대한 매개변수가 모두 있는 생성자 자동 생성  
  그 외 @Builder, @RequiredArgsConstructor, @Slf2j 등 있음

# Exam05

@PathVariable URL 매개변수와 함수 매개변수가 1:1 로 전달되는 변수

* 파라메터 방식 : URL 매개변수를 각각 함수의 매개변수로 전달  
  쿼리스트링과 유사, REST API 프로그램에서 자주 사용됨  
  파라메터 매개변수명 == 함수 매개변수명 (일치)  
  ex) 웹 URL: <http://localhost:8000/exam05/path-variable/name/KangTaeGyung>  
  컨트롤러 URL : @GetMapping("/path-variable//name/{name}")

# Exam06

2개이상 @PathVariable 예제

* @PathVariable *: 함수의 매개변수로 2개 이상 사용 할 수 있음  
  2개 이상 사용 시 파라메터 방식의 각 값과 연결됨*ex) 웹 URL: <http://localhost:8000/exam06/path-variable/id/forbob/name/kang>  
  컨트롤러 URL : @GetMapping("/mutlpath-variable/id/{id}/name/{name}")

# Exam07

@RequestBody Post 방식 데이터 받기, @RestController, @PostMapping

* @RestController : 함수의 리턴값을 JSON 데이터 형식으로 리턴해 주는 컨트롤러  
  ex) Vue + Springboot 프로젝트에서 사용됨
* @PostMapping(URL) : 주로 Insert 요청에 사용되며 http 프로토콜의 body 에 내용을 (json 정보 등) 담아 서버로 전송함
* @RequestBody : POST 방식으로 전송된 json 정보를 컨트롤러에서 함수의 매개변수(객체)로 전달 받을 때 사용됨,   
  주로 Insert 나 큰 데이터(객체 전송) 조회 시 사용되는 어노테이션

# Exam08

ResponseEntity 사용해서 서비스 품질 높이기

* ResponseEntity : frontend로 데이터(body) + 메세지를(header) 함께 전송 할 때 사용  
  상태 메세지를 통해 클라이언트에서는 다양한 처리를 추가적으로 할 수 있음  
  ex) new ResponseEntity<Object>(list, HttpStatus.OK) - 데이터 + 메세지  
  new ResponseEntity<Object>(HttpStatus.BAD\_REQUEST) - 메세지
* 주요 상태 메세지 정보 : 400 번 대 (에러 ), 200 번 대 (성공 )   
  HttpStatus.OK : 성공메세지 (200)  
  HttpStatus.NO\_CONTENT : 서버에 요청데이터 없음(204)  
  HttpStatus .BAD\_REQUEST : 잘못된 요청(400)  
  HttpStatus .INTERNAL\_SERVER\_ERROR : 서버 내부 에러(500)  
  HttpStatus .NOT\_FOUND : 페이지 정보 없음(404)

# Exam09

ResponseEntity 사용해서 서비스 품질 높이기, try ~ catch 같이 사용하기

* ResponseEntity + try ~ catch 문 과 함꼐 사용해서 소스 품질을 높일 수 있음
* Try/Catch + ResponseEntity 주요 구문 예  
  try {  
   결과 = 실행문  
   if(결과) {  
   // 성공  
   return new ResponseEntity<Object>(객체, HttpStatus.OK)  
   } else {  
   // 요청 데이터 없음  
   return new ResponseEntity<Object>(HttpStatus.NO\_CONTENT)  
   }  
  } catch ( Exception e) {  
   // 서버 내부 에러  
   return new ResponseEntity<Object>(HttpStatus. INTERNAL\_SERVER\_ERROR)  
  }

# Exam10

ResponseEntity 에서 내장 함수를 이용해서 클라이언트에 전송하기 #2

* 함수 호출해서 클라이언트에 전송(빌더 패턴) :   
  ex) 성공 : return ResponseEntity.ok().body(list)   
  데이터 없음 : ResponseEntity.noContent().build()  
  에러 : ResponseEntity.status(HttpStatus.BAD\_REQUEST).build()