초격차 패키지 Online.

API 설계

PART1 | 애노테이션 기반 설계

@Controller, @RestController

PART2 | 함수 기반 설계

함수형 프로그래밍으로 만드는 API

PART3 | 요청, 응답의 설계

Handler Methods의 활용

PART4 | @ControllerAdvice

공통 에러 응답을 설계하는 법

PART5 | 컨트롤러 테스트

Spring Web API를 테스트하는 방법

PART6 | TDD 방식으로 복습하기

테스트 주도 개발 과정 맛보기

API 설계

4 @ControllerAdvice

스프링 부트 기본 에러 응답

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Aug 11 08:07:09 KST 2021
There was an unexpected error (type=Internal Server Error, status=500).

BODY pretty


```
{
    timestamp : "2021-08-10T23:06:44.708+00:00",
    status : 500,
    error : "Internal Server Error",
    path : "/"
}
```

스프링 부트 기본 에러 응답

```
server.error.include-exception=true
server.error.include-message=always
server.error.include-stacktrace=always
```

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Aug 11 08:07:48 KST 2021
There was an unexpected error (type=Internal Server Error, status=500).
this is test
java.lang.Exception: this is test
at com.uno.getinline.controller.BaseController.root(BaseController.java:13)
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccesso
at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethoc
at java.base/java.lang.reflect.Method.invoke(Method.java:567)
at org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(Invocab
at org.springframework.web.method.support.InvocableHandlerMethod.invokeForReques

BODY pretty


```
{
    timestamp : "2021-08-10T23:08:08.927+00:00",
    status : 500,
    error : "Internal Server Error",
    exception : "java.lang.Exception",
    trace : "java.lang.Exception: this is test\n\tat com.
    message : "this is test",
    path : "/"
}
```


시스템 정보 유출 조심

BasicErrorController

스프링 부트의 기본 응답이 마음에 든다면

- BasicErrorController 를 상속 받아서
  - 특정 메소드만 오버라이드 하거나
  - 특정 핸들러 메소드를 추가하는 식으로 응용
- BasicErrorController 의 핸들러 메소드
  - BasicErrorController.errorHtml() -> 뷰 응답
  - BasicErrorController.error() -> json body 응답

커스텀 에러 페이지: 기본

간단히 static html 이나 template 파일을 추가해서 커스텀 페이지를 등록하는 법

- 단일 기본 페이지
  - /resources/static/error.html
  - /resources/public/error.html
  - /resources/template/error.[템플릿확장자]
- http status 별 기본 페이지
  - /resources/[static|public|template]/error/{http status 번호}.[html|템플릿확장자]
  - /resources/[static|public|template]/error/4xx.[html|템플릿확장자]
  - /resources/[static|public|template]/error/5xx.[html|템플릿확장자]

@ExceptionHandler

비즈니스 로직이 던진 예외에 반응하는 핸들러 메소드

- 위치: 특정 컨트롤러 클래스 내부 or @ControllerAdvice 컴포넌트 내부
- 특정 예외에 반응
- 예외 처리 범위
  - 컨트롤러 안에 작성했을 경우: 해당 컨트롤러만
  - @ControllerAdvice 에 작성했을 경우: 프로젝트 전체

@ExceptionHandler

핸들러 메소드에 속하기 때문에 입출력 자료형도 핸들러 메소드와 유사하지만,

예외를 입력 인자로 다룰 수 있다는 점이 차이점

- https://docs.spring.io/spring-framework/docs/current/reference/html/web.html#mvc-ann-exceptionhandler-args
- https://docs.spring.io/spring-framework/docs/current/reference/html/web.html#mvc-ann-exceptionhandler-return-values

@ExceptionHandler

```java
@ExceptionHandler
public ResponseEntity<APIErrorResponse> general(RuntimeException e) {
    return ResponseEntity.internalServerError().build();
}
```

```java
@ExceptionHandler(RuntimeException.class)
public ResponseEntity<APIErrorResponse> general(RuntimeException e) {
    return ResponseEntity.internalServerError().build();
}
```

```java
@ExceptionHandler(RuntimeException.class)
public ResponseEntity<APIErrorResponse> general(Exception e) {
    return ResponseEntity.internalServerError().build();
}
```

```java
@ExceptionHandler({RuntimeException.class, IOException.class})
public ResponseEntity<APIErrorResponse> general(Exception e) {
    return ResponseEntity.internalServerError().build();
}
```

@ControllerAdvice

@ExceptionHandler 를 모아서 글로벌하게 적용할 때 쓰는 애노테이션

- 종류
  - @ControllerAdvice
  - @RestControllerAdvice = @ControllerAdvice + @ResponseBody
- 속성
  - value == basePackages
  - basePackages: 적용 범위를 문자열을 이용해 특정 패키지로 지정
  - basePackageClasses: 적용 범위를 대표 클래스 한 개를 이용해 특정 패키지로 지정
    - basePackages 를 type-safe 하게 사용하기 위해 제공하는 옵션
  - assignableTypes: 적용 범위를 특정 클래스에 할당할 수 있는 컨트롤러로 지정
  - annotations: 적용 범위를 특정 애노테이션을 사용한 컨트롤러로 지정

ResponseEntityExceptionHandler

Spring MVC 에서 내부적으로 발생하는 예외들을 처리하는 클래스

- API 예외 처리를 담당하는 @ControllerAdvice 클래스에서 상속 받아 사용
- 커스터마이징을 원하는 특정 메소드를 오버라이드

Reference

- https://docs.spring.io/spring-framework/docs/current/reference/html/web.html#mvc-exceptionhandlers
- https://docs.spring.io/spring-framework/docs/current/reference/html/web.html#mvc-ann-controller-advice