라이브 세션-예외 처리-2022.08.25(수)

지난 시간 학습 리뷰

- Controller 리뷰
- DTO 리뷰
- Mapper 리뷰
- 공지사항(Notice) 기능을 만들어 보면서 리뷰 해 봅시다!

이번 시간 이야기 나눌 내용

체크 예외(Checked Exception)와 언체크 예외(Unchecked Exception)

- 체크 예외
  - 반드시 체크 해야 하는 예외
    - `SQLException`, `ClassNotFoundException`,
- 언체크 예외
  - 캐치(catch)할 필요 없는 예외

Controller에서 @ExceptionHandler 사용의 장/단점

@RestControllerAdvice를 이용한 예외처리

of() 등의 정적 팩토리 메서드(static factory method)

- **정적 팩터리 메서드란?**
  - new를 직접적으로 사용하지 않고, 클래스의 인스턴스를 생성하는 방법
- 정적 팩토리 메서드의 장점
  - **이름을 통해 객체의 의미를 쉽게 알 수 있다.**

    예)
    - Stream.of(1, 2, 3)
      - a stream that is made **of** the numbers 1, 2 and 3
    - List.of(1, 2, 3, 4)
      - a list that is made **of** the numbers 1, 2, 3, and 4
    - `ErrorResponse of(BindingResult bindingResult)`
      - a ErrorResponse instance is made **of** BindingResult object
    - `ErrorResponse of(Set<ConstraintViolation<?>> violations)`
      - a ErrorResponse instance is made **of** BindingResult object

  - **매 번 새로운 객체를 생성할 수 도 있고, 아닐 수 도 있다.**
    - `Boolean.TRUE`
    - 새로운 객체를 생성할 때 제약 조건을 걸 수도 있다.

  - **하위 타입 객체 반환 가능**

```
public class Foo {

    private Foo() {

    }

    public static Foo foo() {
        return new Foo();
    }
```

```
    public static Foo fooBar() {
        return new Bar();
    }

    private static class Bar extends Foo {

        public Bar() {

        }
    };
};
```

- 정적 팩터리 메서드 작성 시점에 반환될 객체의 클래스 존재하지 않아도 됨
  - SPI(Service Provider Interface)에서 많이 사용
  - 예)
    - Class.forName(”com.mysql.jdbc.Driver”)

비즈니스적인 예외 throw/catch

- **백엔드 서버와 외부 시스템과의 연동에서 발생하는 에러 처리**
- **시스템 내부에서 조회하려는 리소스(자원, Resource)가 없는 경우**

Custom Exception 사용

- 우리가 작성한 `BusinessLogicException` 같은 예외 클래스

실습 과제 설명

- 과제 1
- 과제 2
- 과제 3

Stream API 설명