라이브 세션-Spring Data JDBC란-2022.08.26(금)

지난 시간 학습 리뷰

- @ExceptionHandler 리뷰
- @RestControllerAdvice 리뷰
- 비즈니스 로직 예외 처리 리뷰
- Error 전용 Response 클래스 리뷰
    - ErrorResponse 클래스 리뷰

✅ 아바타 실습: 공지사항 기능 구현을 통한 리뷰

: 다 같이 조금만 리뷰해 봅시다. ^^

---

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() {

        }
    };
};
```

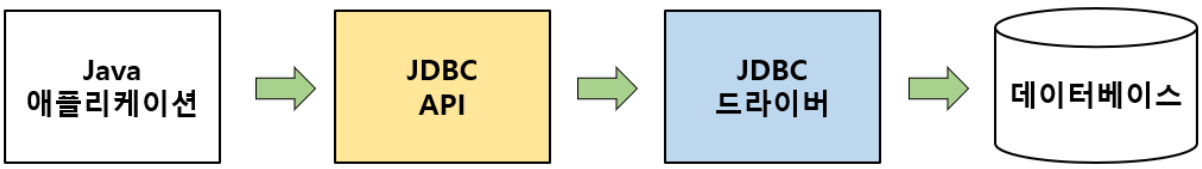
- 정적 팩터리 메서드 작성 시점에 반환될 객체의 클래스 존재하지 않아도 됨
  - SPI(Service Provider Interface)에서 많이 사용
  - 예)
    - Class.forName(”com.mysql.jdbc.Driver”)

✅ In-Memory DB란?

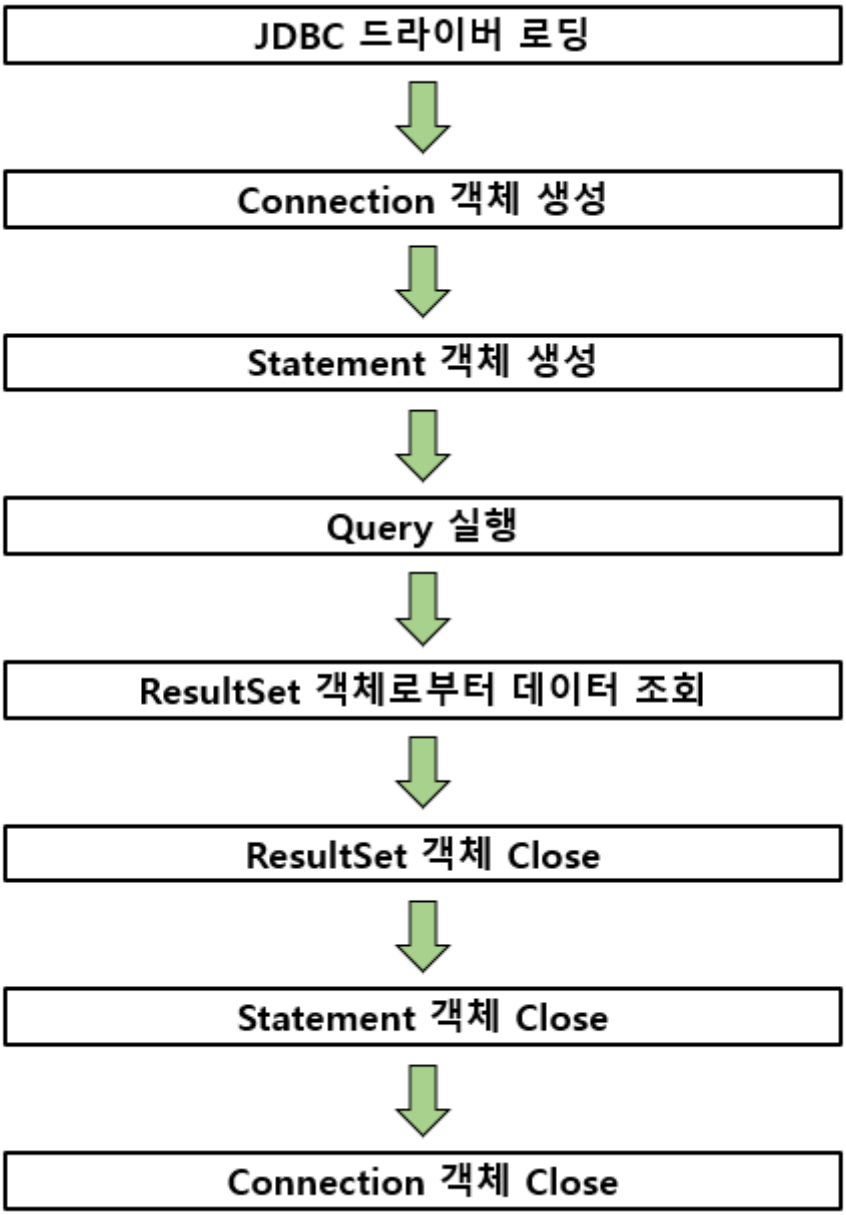
- 메모리 DB
- 애플리케이션이 종료되면 데이터는 사라짐
- ⭐ 로컬 개발 환경에서 테스트 용도로 사용
- 우리가 사용하는 In-Memory DB는 **H2**
  - 설정 방법

✅ JDBC API 리뷰

1️⃣ JDBC의 동작 흐름

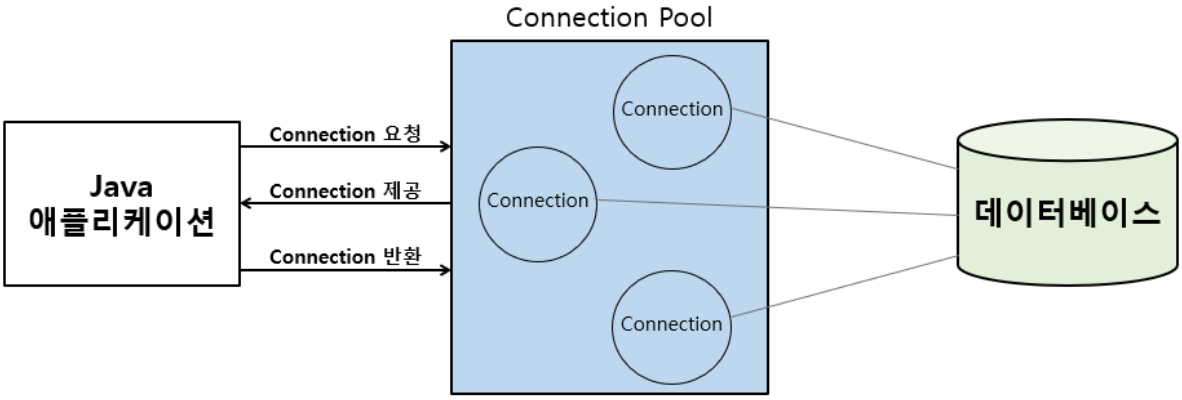


2️⃣ JDBC API 사용 흐름



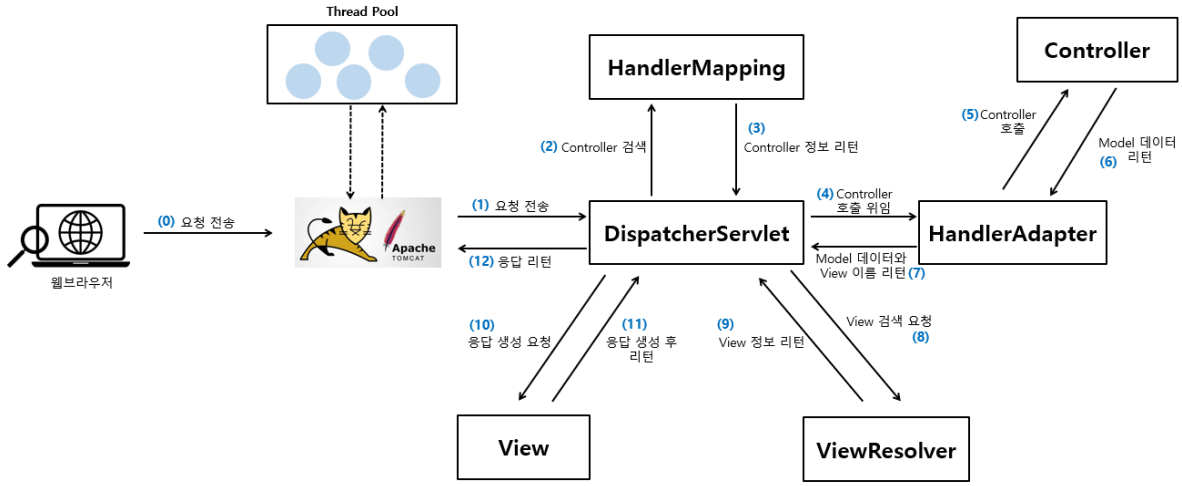
3️⃣ DB Connectioon Pool

- Apache commons DBCP
- HikariCP



* 참고

Thread Pool



HTTP Connection Pool

- Server가 Keep-Alive 지원 필요
- HttpClient
- HttpComponentsClientHttpRequestFactory

✅ Java의 데이터 액세스 기술 종류

1️⃣ SQL 중심 기술

- 순수 JDBC API 이용
- iBatis 또는 myBatis
- Spring JDBC

2️⃣ ORM(Object-Relational Mapping)중심 기술

- Spring Data JDBC
- JPA
- Spring Data JPA

3️⃣ 순수 JDBC API 기술 들여다 보기

- 구현 코드 살짝 들여다 보기

4️⃣ Spring JDBC 들여다 보기

- 구현 코드 살짝 들여다 보기

Hello, Spring Data JDBC 코드 리뷰

Spring Data JDBC 적용 순서

1. `build.gradle`에 사용할 데이터베이스를 위한 의존 라이브러리를 추가합니다.
2. `application.yml` 파일에 사용할 데이터베이스에 대한 설정을 합니다.
3. ‘`schema.sql`’ 파일에 필요한 테이블 스크립트를 작성합니다.
4. application.yml 파일에서 ‘schema.sql’ 파일을 읽어서 `테이블을 생성`할 수 있도록 초기화 설정을 추가합니다.
5. 데이터베이스의 테이블과 매핑할 `엔티티(Entity) 클래스를 작성`합니다.
6. 작성한 엔티티 클래스를 기반으로 데이터베이스의 작업을 처리할 `Repository 인터페이스를 작성`합니다.
7. 작성된 Repository 인터페이스를 `서비스 클래스에서 사용할 수 있도록 DI` 합니다.
8. DI 된 Repository의 메서드를 사용해서 `서비스 클래스에서 데이터베이스에 CRUD 작업을 수행`합니다.