스프링(Spring)과 롬복(Lombok)의 결합, @RequiredArgsConstructor를 사용한 생성자 주입

Lombok에 대한 이해와 활용

롬복이란?

- 어노테이션 기반으로 코드를 자동완성
- Getter, Setter, 생성자 등을 자동완성

```
@Getter
@Setter
public class User {

    private String email;
    private String name;

}
```

@RequiredArgsConstructor

- NotNull, final이 붙은 변수들에 대해 생성자를 만들어주는 기능 제공

```
@RequiredArgsConstructor
public class User {

    private final String email;
    private final String name;

   /* 이 생성자를 자동으로 만들어줌
   public class User(String email, String name) {
        this.email = email;
        this.name = name;
   }
   */

}
```

Spring 프레임워크와 Lombok의 결합

Spring 프레임워크를 통한 생성자 주입

```
@Service
public class UserService {
```

```
    private UserRepository userRepository;
    private PasswordEncoder passwordEncoder;

    @Autowired
    public UserService userService(UserRepository userRepository,
PasswordEncoder passwordEncoder) {
        this.userRepository = userRepository;
        this.passwordEncoder = passwordEncoder;
    }

}
```

final 키워드와의 결합

- 위의 생성자에서 UserRepostory와 PasswordEncoder은 Spring 컨테이너가 관리하는 싱글톤 객체이기에 변할 일이 없음
- 생성자를 통해 주입되는 시점에 불변성을 보장하는 final 키워드를 붙임

```
@Service
public class UserService {

    private final UserRepository userRepository;
    private final PasswordEncoder passwordEncoder;

    @Autowired
    public UserService userService(UserRepository userRepository,
PasswordEncoder passwordEncoder) {
        this.userRepository = userRepository;
        this.passwordEncoder = passwordEncoder;
    }

}
```

Lombok과의 결합

- 생성자 주입 코드를 모든 빈마다 생성하려면 번거로움.
- 새로운 빈을 생성자 주입으로 추가하는 과정도 번거로움

> 스프링 프레임워크는 1개 뿐인 생성자의 경우 @Autowired를 자동으로 인식해 처리

```
@Service
@RequiredArgsConstructor
public class UserService {

    private final UserRepository userRepository;
    private final PasswordEncoder passwordEncoder;

}
```