WAFFLE

STUDIO

와플스튜디오 Backend Seminar

Instructor: 강지혁

2022.10.04.(화) 19:30

Session 2

Table of Contents

WA#LE
STUDIO

Review

우리가 한 일

- 서버, HTTP, 스프링 이해하기
- 필요한 컴포넌트 (Bean) 만들고, 사용하는 법 이해하기
- 요청 받고, 적절히 처리하기
- 관계형 데이터베이스
- Spring-Data-JPA
- 모델 설계하기

WA#LE
STUDIO

Spring.. 이제 잘 아는 걸까?

이제 여러분이 할 수 있는 것

- @RequestMapping 바탕으로 HTTP 요청 받기
- Service 레이어에서 요청을 처리하기
- Exception Handling
- DB에서 읽어오기, DB에 저장하기

어노테이션은 정확히
어떻게 처리되는 걸까?

컨트롤러, 서비스는 각자
어떤 책임을 가지는 걸까?

예외 처리되면 트랜잭션
은
어떻게 되는거지?

복잡한 쿼리는 어떻게 짜야 되는거야..

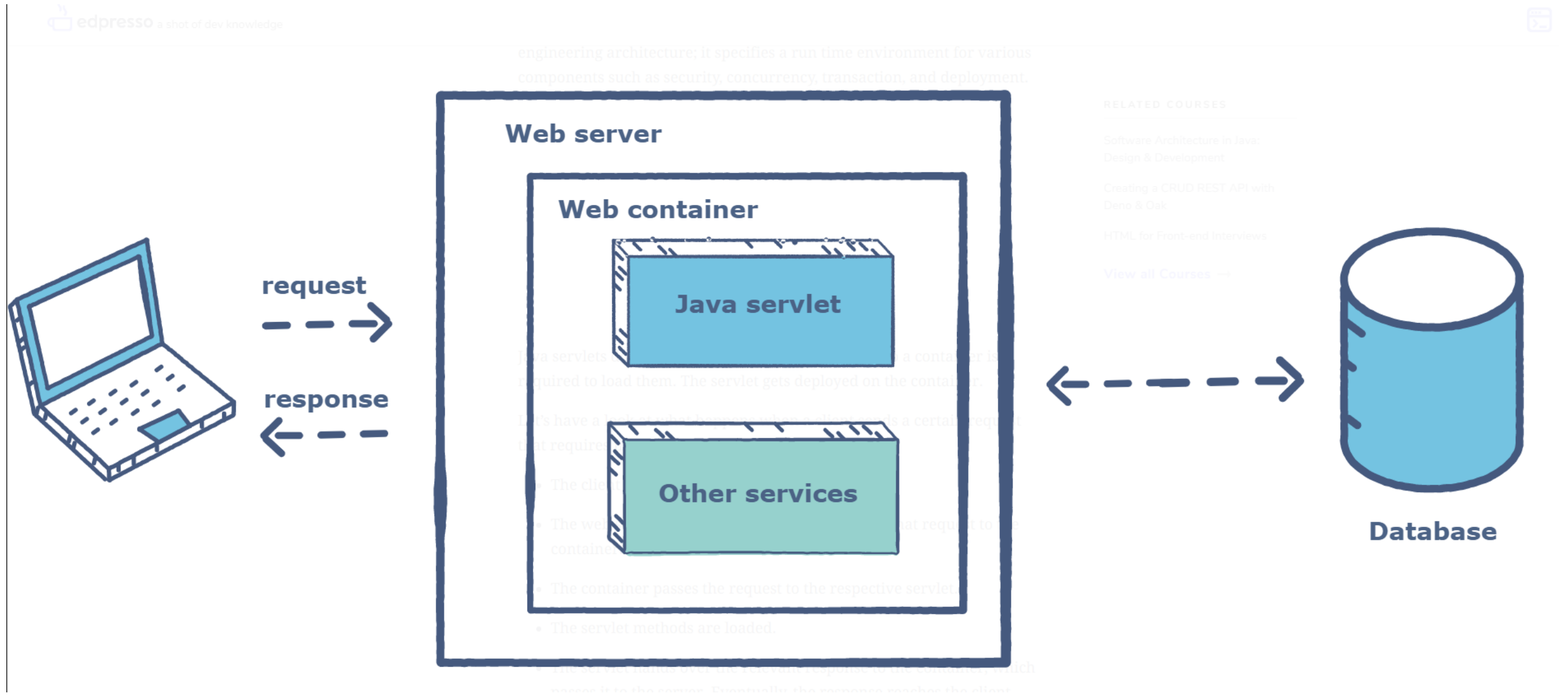
서버에 인증 로그인은 어떻게 하는거지?

WA#LE
STUDIO

Spring : 서블릿 컨테이너

JVM에서 동적인 요청을 처리하는 방법

하나의 요청은 곧 하나의 스레드에, Servlet 객체로 치환됨



Spring : 서블릿 컨테이너

No-Magic

package javax.servlet
package javax.servlet.http

```
public interface Servlet {

    public void init(ServletConfig config) throws ServletException;

    public ServletConfig getServletConfig();

    public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException;

    public String getServletInfo();


    public void destroy();
}
```

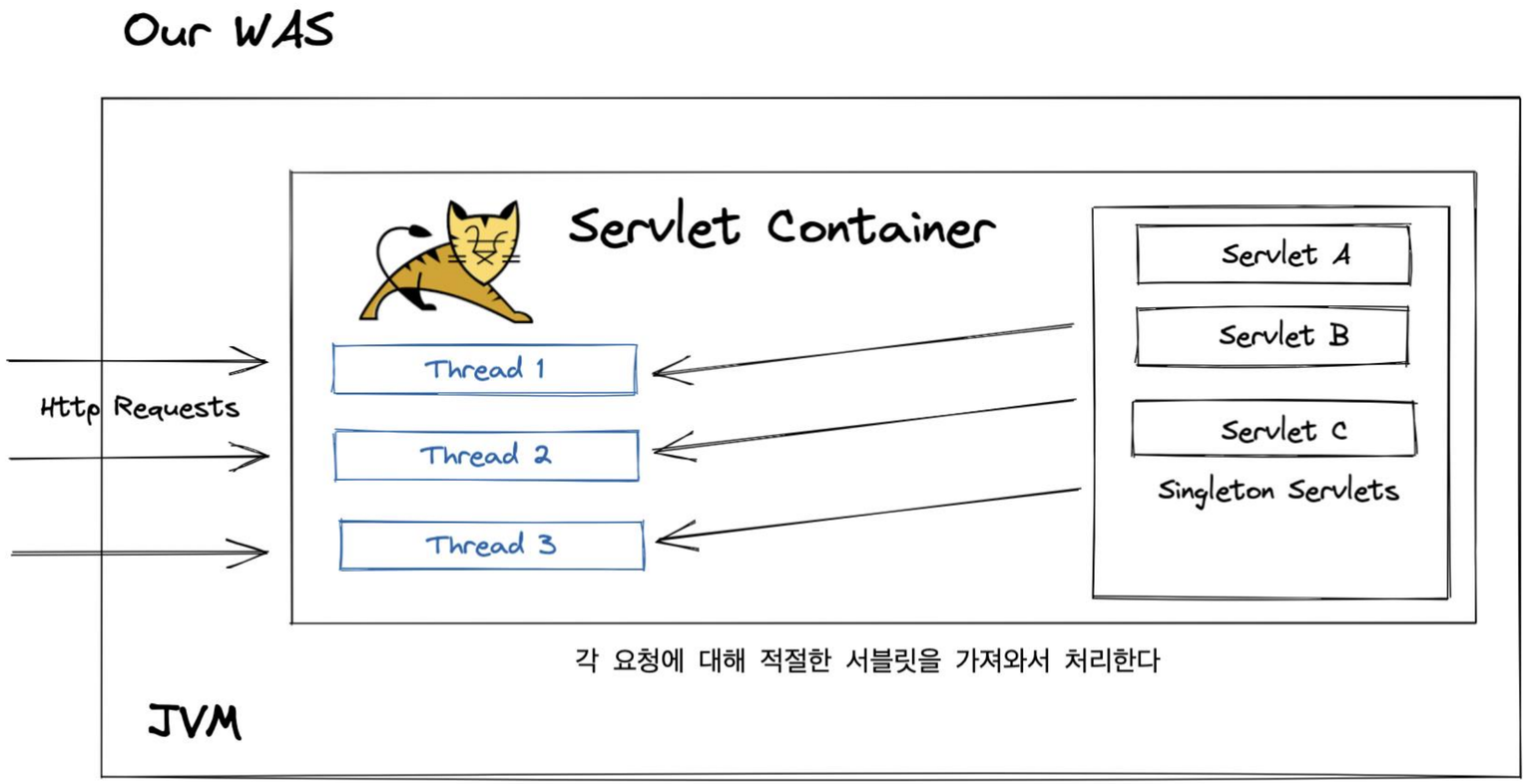
WA LE
STUDIO

Spring : 서블릿 컨테이너

- Servlets are Singleton
- Resides at Servlet Container

Spring Uses: Embedded Apache Tomcat

```
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080
```

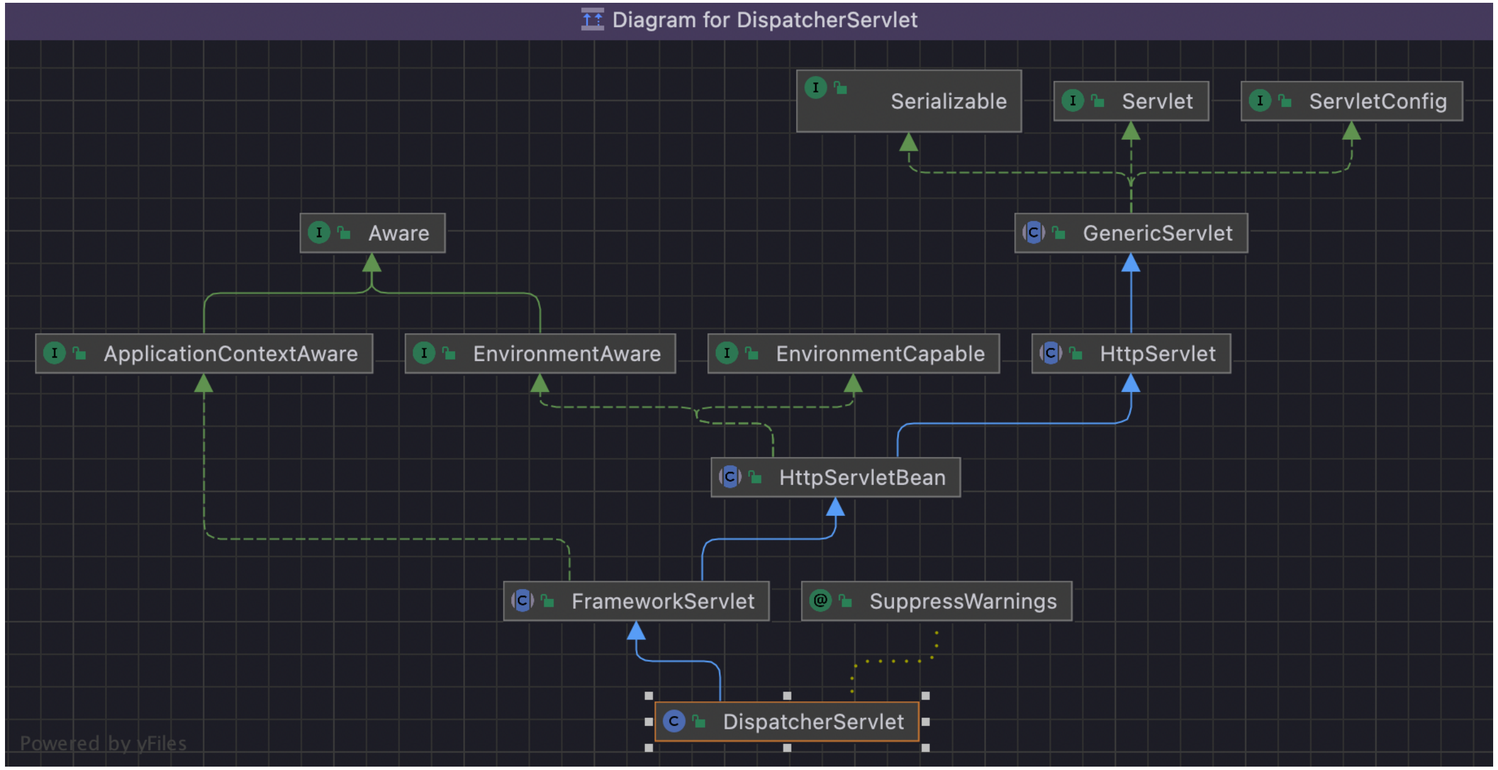


WA LE STUDIO

Spring : 서블릿 컨테이너

스프링에서는,,

DispatcherServlet 하나면 충분합니다.



Cmd + Alt + U : 상속 트리 보기

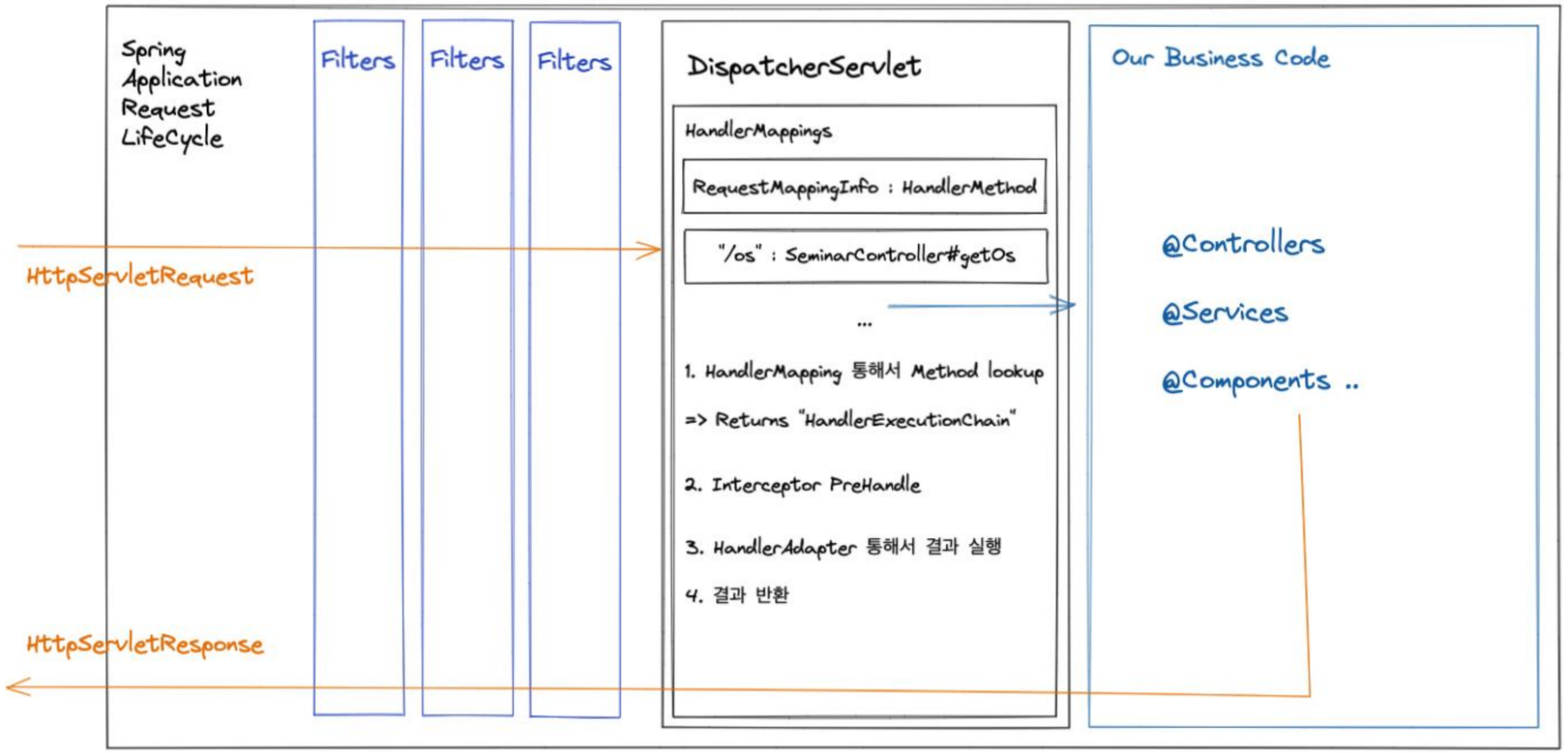
Ctrl + H : 상속 계층 보기

WA LE STUDIO

Spring : 요청 처리 흐름

```
@Override
protected void doService(HttpServletRequest request, HttpServletResponse response) throws Exception
---
```

우리는 요청의 흐름에 어디까지 관여할 수 있을까?



WA LE
STUDIO

Authorization

로그인.. 어떻게 구현할까?

- Username & Password 입력
- 소셜 로그인

인증 정보는 어떻게 보관할까?

- Cookie, Session, JWT
  - 쿠키 : set-cookie를 통해 인증 정보를 클라이언트가 가지고 있도록 함
  - 세션 : 서버에서 클라이언트의 연결 정보를 저장하고 있음
  - JWT : 데이터 위변조 검증 가능, 우수한 확장성, STATELESS 통신
- (Reference)

WA#LE STUDIO

Authorization

JWT : Json Web Token

- Header, Payload, Signature로 구성
- 클라이언트는 최초 요청에 대해 토큰 발급을 요청
- 이후의 통신에서는 토큰을 통해 인증을 처리한다.

JWT는 어디에 담아 보낼까?

- Authorization Header
- 우리는 토큰을 어떻게 받아오고, 인증 처리할 수 있을까?
  - @RequestHeader?

JWT가 탈취당하면 어쩌지? (ref)

WA LE
STUDIO

Authorization

JWT : Json Web Token

- Header, Payload, Signature로 구성
- 클라이언트는 최초 요청에 대해 토큰 발급을 요청
- 이후의 통신에서는 토큰을 통해 인증을 처리한다.

JWT는 어디에 담아 보낼까?

- Authorization Header
- 우리는 토큰을 어떻게 받아오고, 인증 처리할 수 있을까?
  - @RequestHeader?

JWT가 탈취당하면 어쩌지? (ref)

WA LE
STUDIO

Authorization

실습 타임

스켈레톤 코드를 바탕으로,

인증이 필요한 요청을 식별하고 & 인증 정보를 컨트롤러에게 넘겨봅시다.

“일반적인” 인증 방법 :

OAuth2, Spring-Security, 별도의 인증 서버, 게이트웨이, etc ...

WA#LE
STUDIO

Spring Data – @Transactional

우리가 트랜잭션을 만들기 위해서는..

```
start transaction;
insert into user (id, name, age, gender) values (0, 'asdf', 1, 'MALE');
rollback;
```

스프링에서는 !?

@Transactional Annotation To Rescue

또 알아서 해줘? 🦁♂ 🦁♂ 🦁♂

그럼 그냥 가져다 쓰면 되겠네 !? 🤦♂ 🤦♂ 🤦♂

WA LE
STUDIO

Spring Core : PSA

Portable Service Abstraction

여러분의 뇌빼기 코딩을 돕는 스프링의 핵심 Value

왜 스프링을 쓰고, 왜 “객체 지향” 프레임워크를 쓰는가?

변경에 대해 유연하게 대응하고 싶다. 즉, 구현 세부사항에 의존하고 싶지 않다 !!

어떻게 이걸 가능하게 했을까?

사용할 때 어떤 옵션들을 제공할까?

WA#LE
STUDIO

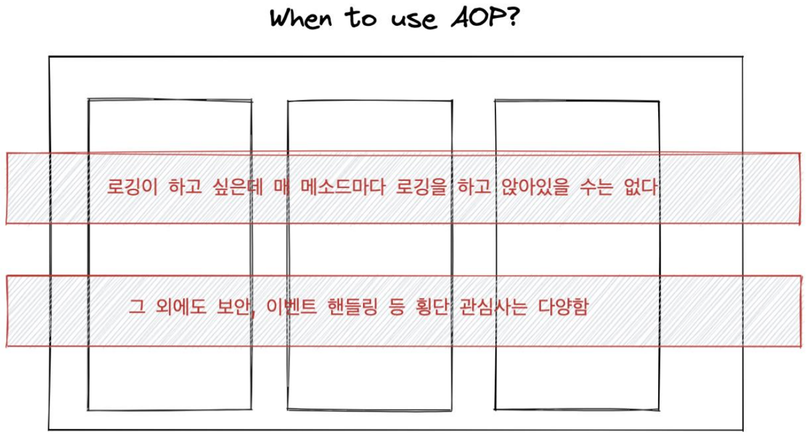
Spring Core : AOP, CGLIB, Proxy Pattern

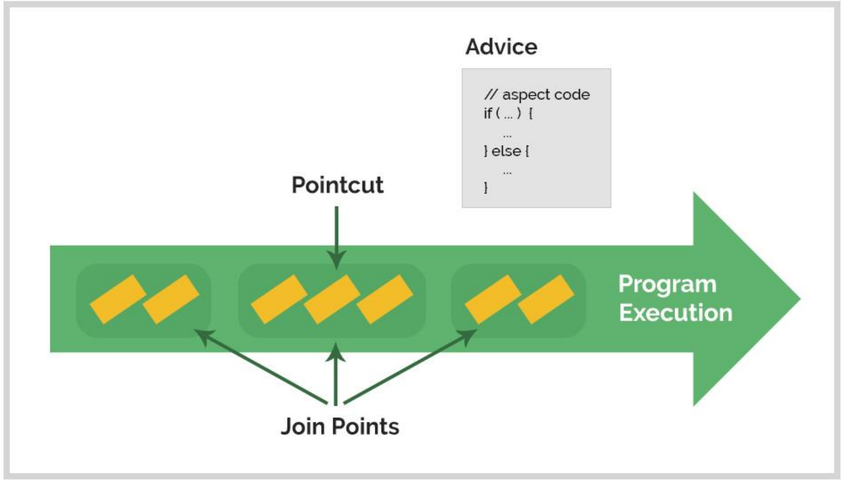
AOP (Aspect Oriented Programming), “관점” 중심의 프로그래밍

- 횡단 관심사에 대한 일괄 처리
- CGLIB를 통한 프록시 패턴

@Aspect 어노테이션을 통해 커스텀해보자!

- @Around, @Before, @After,,





Spring Core : AOP (ref)

Spring AOP use-case: (ref)

WAFFLE STUDIO

Spring Core : AOP, CGLIB, Proxy Pattern

CGLIB

프록시를 만드는 방법 중 하나

Runtime-weaving, 아래와 같이 트랜잭션 여닫는 코드 넣어줌

```
@OurService
class OurService {

    @Transactional
    fun a() {
      // ..
    }

}
```

```
class OurServiceProxy: OurService {

    override fun a() {
      // 1. start transaction

      // 2. execute
      super.a()

      // 3. commit & rollback
    }

}
```

CGLIB Baeldung (ref)

WA LE
STUDIO

Spring Core : Factory Bean, Bean Factory, ApplicationContext

오케이, 프록시 쓰는구나.

언제 만들까?

package org.springframework.aop.framework

public class ProxyFactoryBean

FactoryBean<T> : Bean을 정의하는 또다른 방법

@Transactional 어노테이션은 결론적으로..

Spring Tx Deep Dive : (ref) **<< 이런 과정을 거쳐서 완성된다**

Spring Core : Factory Bean (ref)

What's a Factory Bean? (ref)

Which Interceptor Handles @Transactional

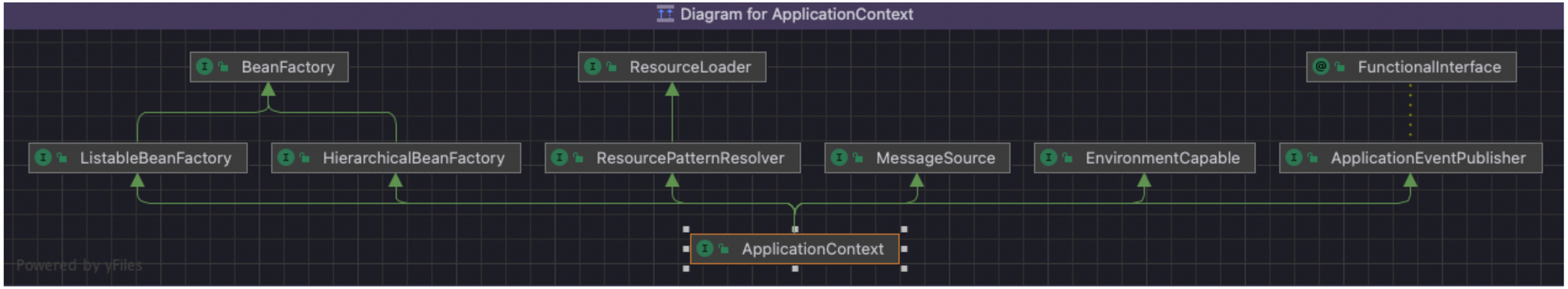
WA LE STUDIO

Spring Core : Bean Factory, ApplicationContext, IOC

Bean은 그냥 Java Singleton 객체 : FactoryBean은 Bean을 찍어내는 특이한 Bean

Bean을 스프링이 주입해주는 것 : Inversion Of Control

스프링은 ApplicationContext 라는 IoC 컨테이너를 통해 애플리케이션을 관리



다시 한번, No-Magic ! 😊

Spring ApplicationContext : (ref)

WA#LE STUDIO

Spring Data – DataSource Configuration

application.yaml

```yaml
spring:   You, Moments ago • Uncommitted changes
  profiles:
    active: dev
  datasource:
    url: jdbc:mysql://localhost:3306/seminar?serverTimezone=UTC
    username: root
    password: seminar
  jpa:
    properties:
      hibernate:
        show_sql: true
    hibernate:
      ddl-auto: update
```

More On 예제!

특별히 설정 값을 넣은 Bean을 생성해주지 않아도..

설정파일의 값들을 읽어올 수 있도록 도와주는 것은

Spring-Boot의 _____ 덕분입니다 ☺

WA LE
STUDIO

Spring Data – QueryDSL, RDB JOIN

RDB JOIN : 기본기!

SQL JOIN (ref)

Remind : RDB를 쓰면서...

우리는 중복을 피하고 싶었음 => 데이터를 분리하고, id (foreign key)를 통해 참조하기 시작

읽어올 때는 !?

- **Player # 100 의 이름은 손흥민이고, 이 사용자의 소속팀은 Club # 34531 입니다.** 🤦‍♂️
- **Player # 100 의 이름은 손흥민이고, 이 사용자의 소속팀은 토트넘입니다.** 🦁♂️

```
select * from users inner join lecture_entity le on users.id = le.instructor_id;
```

예시

WA LE STUDIO

Spring Data – QueryDSL, RDB JOIN

QueryDSL 사용하기

- Q-Class라고 하는 객체를 통해 SQL 매핑

QueryDSL Introduction (ref)

- Fluent API
- 복잡한 쿼리를 짜야 할 때 👀
- 쿼리를 여러 번 쪼개기 vs Join 적절히 활용하기

WA#LE
STUDIO

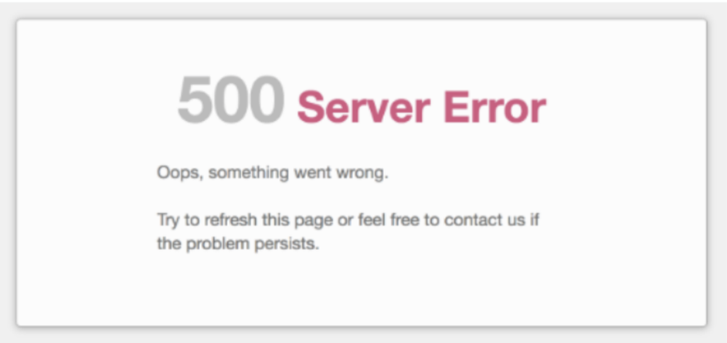
Wrap It UP

Re-Remind

500을 피하자

INTERNAL SERVER ERROR

- logic상 status code 500이 발생할 수 있는 여지는 최대한 없애기
- 모든 프로그래밍이 그렇지만, 서버 프로그래밍은 염려와 강박을 탑재할 것
- 안정적인 서버 자체가 서비스의 가치
- 유저가 서버의 존재를 느낄 일이 없어야



- 문제될 상황에 대해선 합리적인 처리가 중요
  - 예상되는 원인을 코드상에 드러나게 짚어서 대응
  - 문제의 책임과 원인을 찾고 외부에 알리는 프로그래밍

WA#LE STUDIO

Q&A

WAFFLE STUDIO

WAFFLE STUDIO