МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНЙ УНІВЕРСИТЕТ

Кафедра програмного забезпечення автоматизованих систем

ЗВІТ

про виконання

Лабораторної роботи №3

з дисципліни "Розробка Web-додатків на Java з допомогою Spring фреймворку"

Перевірив роботу: Виконав роботу:

"\_\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ студент гр. ПЗС-2244

викладач\_\_\_\_\_\_\_\_\_\_\_\_\_ Брус В.М.

Черкаси 2023

Тема: Spring

Мета: Реалізувати клас аспект, методи advice, pointcut

Завдання:

У проекті який створювали у 1-2 л.р. необхідно реалізувати:

1. Мінімум 1 Aspect клас

2. Мінімум 3 Advice методи

3. Мінімум 3 Pointcut

4\*(не обов'язкове). 1 комбінований Pointcut

Хід роботи:

1. Видалення файлу spring-config.xml:

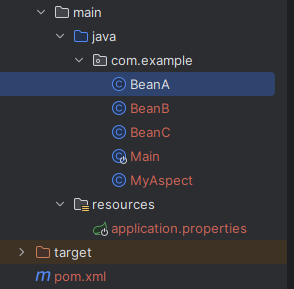


Рисунок 1 – Структура файлів

1. Реалізація класу аспект, методів advice, pointcut.
2. Результат роботи програми в терміналі:



Рисунок 2 – Вивід в термінал

Лістинг

Клас BeanA.java:

package com.example;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class BeanA {

private final BeanB beanB;

@Autowired

public BeanA(BeanB beanB) {

this.beanB = beanB;

}

public void doSomething() {

beanB.doSomething();

}

}

Клас BeanB.java:

package com.example;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class BeanB {

private BeanC beanC;

@Autowired

public void setBeanC(BeanC beanC) {

this.beanC = beanC;

}

public void doSomething() {

beanC.doSomething();

}

}

Клас BeanC.java:

package com.example;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

import javax.annotation.PostConstruct;

import javax.annotation.PreDestroy;

@Component

public class BeanC {

@Value("${my.property}")

private String property;

@PostConstruct

public void init() {

System.out.println("BeanC is initialized with property: " + property);

}

@PreDestroy

public void destroy() {

System.out.println("BeanC is being destroyed");

}

public void doSomething() {

System.out.println(property);

}

}

Клас Main.java:

package com.example;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class Main {

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(Main.class, args);

BeanA beanA = context.getBean(BeanA.class);

beanA.doSomething();

}

}

application.properties:

my.property=Hello, World!

MyAspect.java:

package com.example;  
  
import org.aspectj.lang.annotation.After;  
import org.aspectj.lang.annotation.Aspect;  
import org.aspectj.lang.annotation.Before;  
import org.aspectj.lang.annotation.Pointcut;  
  
@Aspect  
public class MyAspect {  
  
 @Pointcut("execution(\* com.example.BeanA.doSomething(..))")  
 public void beanAPointcut() {}  
  
 @Pointcut("execution(\* com.example.BeanB.doSomething(..))")  
 public void beanBPointcut() {}  
  
 @Pointcut("execution(\* com.example.BeanC.doSomething(..))")  
 public void beanCPointcut() {}  
  
 @Before("beanAPointcut()")  
 public void beforeBeanAAdvice() {  
 System.out.println("Before BeanA's doSomething method");  
 }  
  
 @After("beanBPointcut()")  
 public void afterBeanBAdvice() {  
 System.out.println("After BeanB's doSomething method");  
 }  
  
 @Before("beanCPointcut()")  
 public void beforeBeanCAdvice() {  
 System.out.println("Before BeanC's doSomething method");  
 }  
  
 @Pointcut("beanAPointcut() && beanBPointcut()")  
 public void combinedPointcut() {}  
  
 @Before("combinedPointcut()")  
 public void beforeCombinedAdvice() {  
 System.out.println("Before BeanA and BeanB doSomething methods");  
 }  
}

Висновки

Було створено Aspect клас з трьома Advice методами та трьома Pointcut. Це дозволило додати до проекту додаткову логіку без зміни основного коду.