LAPORAN HASIL PRAKTIKUM 2
PEMROGRAMAN BERORIENTASI OBJEK LANJUT
" (Spring-2) "

Dosen Pengampu :

Sri Hartati Wijono, M.Kom.

Oleh

Nama : Maria Gresia Plena Br Purba

NIM : 235314094

Kelas : DP

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS SANATA DHARMA

YOGYAKARTA

2024

A. TUJUAN

1. Memahami penerapan **IoC** dan **DI** dalam skenario yang lebih kompleks.
2. Mengimplementasikan berbagai jenis **Injection** (Constructor Injection, Setter Injection, Field Injection).
3. Memahami dan menerapkan **Scopes (Singleton vs Prototype)**.
4. Mengelola lifecycle bean menggunakan **init** dan **destroy methods**.
5. Menggunakan anotasi **@Component**, **@Autowired**, dan **@Qualifier** untuk **Dependency Injection (DI)**.
6. Menggunakan **Stereotype Annotations**: **@Controller**, **@Service**, dan **@Repository**

B. PELAKSANAAN PRAKTIKUM

1. Capture code class LibraryController

```java
package com.example.library.managment.controller;

import com.example.library.managment.model.Book;
import com.example.library.managment.service.BookService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;

@Controller
public class LibraryController {

    @Autowired
    private BookService bookService;

    public void addNewBook(String isbn, String title, String author) {
        Book book = new Book(isbn, title, author);
        bookService.addBook(book);
    }

    public void displayAllBooks() {
        for (Book book : bookService.getAllBooks()) {
            System.out.println(x: book);
        }
    }
}
```

Analisis : pada class LibraryController terdapat import Autowired dan Controller. Method addNewBook() berfungsi untuk membuat objek pada class Book dengan nama book yang memiliki parameter berisi isbn, title, author. Setelah objek Book dibuat, method ini akan memanggil method addBook() dengan parameter book yang akan disimpan ke dalam objek bookService untuk menambahkan buku baru. Method displayBooks() berfungsi untuk mengembalikan dann menampilkan daftar buku. Pada method ini, terdapat pemanggilan bookService.getAllBooks() yang berfungsi untuk mendapatkan buku-buku yang ada. Lalu, akan melakukan perulangan untuk mencetak informasi dari buku.

2. **Capture code class Book**

```java
package com.example.library.managment.model;

public class Book {

    private String isbn;
    private String title;
    private String author;

    public Book(String isbn, String title, String author) {
        this.isbn = isbn;
        this.title = title;
        this.author = author;
    }

    public String getIsbn() {
        return isbn;
    }

    public void setIsbn(String isbn) {
        this.isbn = isbn;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    @Override
    public String toString() {
        return "Book [ISBN=" + isbn + ", Title=" + title + ", Author=" + author + "]";
    }
}
```

Analisis : terdapat beberapa method dengan setter dan getter. Getter berfungsi untuk mengembalikan nilai dari variabel-variabel, sedangkan setter untuk mengubah nilai dari variabel. Kemudian, terdapat method toString() yang akan mengembalikan informasi dari buku berupa isbn, title, dan author yang telah disimpan ke dalam variabel yang ada.

3. **Capture code BookRepository**

```java
package com.example.library.managment.repository;

import com.example.library.managment.model.Book;
import java.util.List;

public interface BookRepository {
    void save(Book book);
    List<Book> findAll();
}
```

Analisis : interface ini memiliki method save() dengan isian parameter book yang bertipe Book. Method ini berfungsi untuk menyimpan ataupun menambahkan buku baru. Lalu, terdapat method findAll() dalam bentuk List<Book>yang berfungsi untuk mengambil dan mengembalikan buku-buku yang ada dalam bentuk List<Book>.

4. **Capture code class BookRepositoryImpl**

```java
package com.example.library.managment.repository;

import com.example.library.managment.model.Book;
import java.util.ArrayList;
import java.util.List;
import org.springframework.stereotype.Repository;

@Repository
public class BookRepositoryImpl implements BookRepository {

    private List<Book> books = new ArrayList<>();

    @Override
    public void save(Book book) {
        books.add(e: book);
        System.out.println("Book saved: " + book);
    }

    @Override
    public List<Book> findAll() {
        return books;
    }
}
```

Analisis : class ini mengimplementasikan interface BookRepository. Pada class ini terdapat variabel books yang bertipe List<books> yang digunakan untuk menyimpan daftar buku dalam bentuk implementasi ArrayList<>. Kemudian, terdapat method save() dengan parameter book yang bertipe Book. Method ini akan memanggil method add pada objek books dengan parameter berisi book dan mencetak pesan jika buku tersimpan (“Book saved”). Method findAll() akan mengambil dan mengembalikan buku-buku yang ada setelah disimpan ke dalam objek Book.

5. **Capture code BookService**

```java
package com.example.library.managment.service;

import com.example.library.managment.model.Book;
import java.util.List;

public interface BookService {

    void addBook(Book book);

    List<Book> getAllBooks();
}
```

Analisis : interface ini memiliki method addBook() dengan parameter book yang bertipe Book. Method ini berfungsi untuk menambahkan objek Book baru. Method getAllBooks() akan mengambil dan mengembalikan buku-buku yang ada setelah disimpan ke dalam objek Book dalam bentuk List<Book>.

.

6. **Capture code class BookServiceImpl**

```java
package com.example.library.managment.service;

import com.example.library.managment.model.Book;
import com.example.library.managment.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public class BookServiceImpl implements BookService{

    @Autowired
    private BookRepository bookRepository;

    @Override
    public void addBook(Book book) {
        bookRepository.save(book);
    }

    @Override
    public List<Book> getAllBooks() {
        return bookRepository.findAll();
    }
}
```

Analisis : class ini mengimplementasikan interface BookService. Mendeklarasikan variabel bookRepository dengan tipe BookRepository yang bersifat private. Method addBook() berfungsi untuk menambahkan buku baru. Buku yang ditambahkan akan disimpan menggunakan method save() dari bookRepository. Method getAllBooks() memanggil findAll() yang berfungsi untuk mengambil semua buku dan mengembalikannya dalam bentuk List<Book>.

7. **ApplicationContext.xml**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd">

    <!-- Aktifkan scanning komponen -->
    <context:component-scan base-package="com.example.library.managment" />
</beans>
```

Analisis : xml ini merupakan konfigurasi dari Spring yang menggunakan context:component-scan untuk merujuk kepada package “com.example-library.managment”. Sehingga dapat mendaftarkan secara otomatis semua bean yang ada.

8. **Capture code class App**

```java
package com.example.library.managment;

import com.example.library.managment.controller.LibraryController;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {

    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext(configLocation: "ApplicationContext.xml");
        LibraryController libraryController = context.getBean(type:LibraryController.class);

        libraryController.addNewBook(isbn:"978-1234567897", title:"Spring Framework", author:"Craig Walls");
        libraryController.addNewBook(isbn:"978-9876543210", title:"Java for Beginners", author:"James Gosling");

        System.out.println(x:"\nAll books in library:");
        libraryController.displayAllBooks();
    }
}
```

Analisis : class ini mengimplementasikan penggunaan dari ApplicationContext yang digunakan untuk mengkonfigurasi “ApplicationContext.xml” dengan menggunakan ClassPathXmlApplicationContext. Lalu, menggunakan getBean() untuk mengambil bean LibraryController. Memanggil method addNewBook() pada objek libraryController dan mengisi data pada variabel yang ada di parameternya untuk menambahkan buku baru. Kemudian, menampilkan pesan dan memanggil method displayAllBooks() pada objek libraryController untuk menampilkan semua buku yang ada.

9. **Pom.xml**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>library-managment</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.10</version>
            <type>jar</type>
        </dependency>
    </dependencies>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>17</maven.compiler.source>
        <maven.compiler.target>17</maven.compiler.target>
        <exec.mainClass>com.example.library.managment.LibraryManagment</exec.mainClass>
    </properties>
</project>
```

Analisis : dependencies berfungsi untuk mendeklarasikan pustaka eksternal, yaitu groupId, artifactId, dan version. Lalu, pada properties terdapat encoding, versi dari java yang digunakan, dan mainClass yang akan dijalankan. Pom ini berfungsi untuk memudahkan pengelolaan proyek.

10. **Capture Output :**

```
--- exec:3.1.0:exec (default-cli) @ library-managment ---
Book saved: Book [ISBN=978-1234567897, Title=Spring Framework, Author=Craig Walls]
Book saved: Book [ISBN=978-9876543210, Title=Java for Beginners, Author=James Gosling]

All books in library:
Book [ISBN=978-1234567897, Title=Spring Framework, Author=Craig Walls]
Book [ISBN=978-9876543210, Title=Java for Beginners, Author=James Gosling]
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  2.701 s
Finished at: 2024-09-20T11:13:29+07:00
------------------------------------------------------------------------
```