**WEEK -3(Spring Core, Maven, Spring Data JPA)**

**PART 1: Spring & Maven**

**Hands-on 1**: **Configuring a Basic Spring Application**

* ***BookRepository.java:***

package com.library.repository;

public class BookRepository {

public void printRepo() {

System.out.println("BookRepository: Created A new Repository \n");

}

}

* ***BookService.java:***

package com.library.service;

public class BookService {

public void printService() {

System.out.println("BookService: Created a new Service\n ");

}

}

* ***Main.java:***

package com.library;

import com.library.repository.BookRepository;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

ApplicationContext context =

new ClassPathXmlApplicationContext("applicationContext.xml");

BookService bookService = context.getBean("bookService", BookService.class);

bookService.printService();

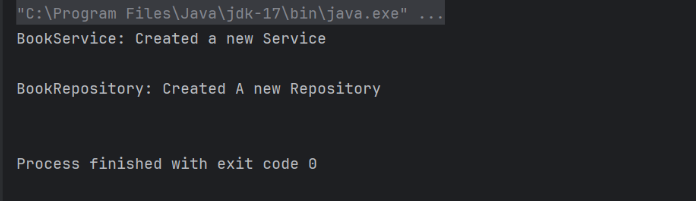
BookRepository bookRepository=context.getBean("bookRepository",BookRepository.class);

bookRepository.printRepo();

}

}

Output:



**Hands-on 2: Implementing Dependency Injection**

* ***BookRepository.java:***

package com.library.repository;

public class BookRepository {

public void accessRepository() {

System.out.println("BookRepository: accessing the database...");

}

}

* ***BookService.java:***

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void performService() {

System.out.println("BookService: handling business logic...");

bookRepository.accessRepository();

}

}

* ***Main.java:***

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

ApplicationContext context =

new ClassPathXmlApplicationContext("applicationContext.xml");

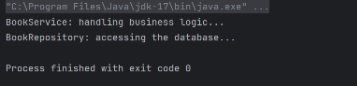
BookService bookService = context.getBean("bookService", BookService.class);

bookService.performService();

}

}

Output:



**Hands-on 3: Creating and Configuring a Maven Project**

* ***BookRepository.java:***

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

public void save(String title) {

System.out.println("BookRepository: Saved " + title);

}

}

* ***BookService.java:***

package com.library.service;

import com.library.repository.BookRepository;

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

import org.springframework.stereotype.Service;

@Service

public class BookService {

private BookRepository bookRepository;

@Autowired

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

bookRepository.save(title);

System.out.println("BookService: Added book " + title);

}

}

* ***LogginAspect.java:***

package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.springframework.stereotype.Component;

@Aspect

@Component

public class LoggingAspect {

@Around("execution(\* com.library.service.\*.\*(..))")

public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

long start = System.nanoTime();

Object result = joinPoint.proceed();

long end = System.nanoTime();

System.out.println("[AOP] " + joinPoint.getSignature()

+ " executed in " + (end - start) / 1\_000\_000 + " ms");

return result;

}

}

* ***Main.java:***

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

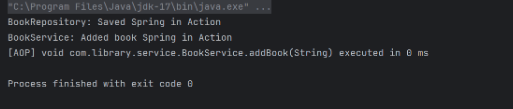
BookService bookService = context.getBean(BookService.class);

bookService.addBook("Spring in Action");

}

}

Output:



**PART 1: Spring-Jpa-Hibernate**

**Hands-on 4:Spring Data JPA – Quick Example**

* ***Country.java:***

package com.cognizant.ormlearn.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="country")

public class Country {

@Id

@Column(name="co\_code")

private String code;

@Column(name="co\_name")

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = name;

}

public void setCode(String code) {

this.code = code;

}

public String getCode() {

return code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

* ***CountryRepository.java:***

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

* ***CountryService.java:***

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

@Transactional

public Country getCountryByCode(String code) {

return countryRepository.findById(code).orElse(null);

}

@Transactional

public Country saveCountry(Country country) {

return countryRepository.save(country);

}

@Transactional

public void deleteCountry(String code) {

countryRepository.deleteById(code);

}

}

* ***OrmLearnApplication.java:***

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import java.util.List;

@SpringBootApplication

public class OrmLearnApplication {

private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);

private static CountryService countryService;

public static void main(String[] args) {

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

countryService = context.getBean(CountryService.class);

LOGGER.info("Inside main");

testGetAllCountries();

}

private static void testGetAllCountries() {

System.out.println("=== All Countries ===");

List<Country> countries = countryService.getAllCountries();

for (Country country : countries) {

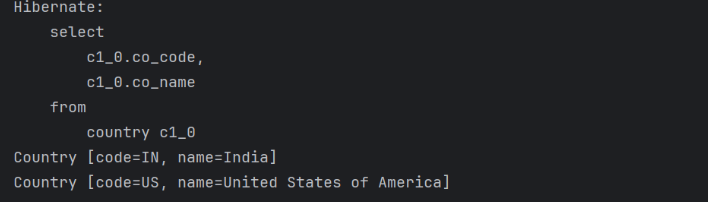
System.out.println(country);

}

}

}

Output:



**Hands-on 5:Difference between JPA, Hibernate and Spring Data JPA**

* ***Employee.java:***

package com.cognizant.Employee;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name="employee")

public class Employee {

@Id

@Column(name="emp\_id")

private Integer id;

@Column(name="emp\_name")

private String name;

@Column(name="emp\_salary")

private Double salary;

public Employee() {}

public Employee(Integer id, String name, Double salary) {

this.id = id;

this.name = name;

this.salary = salary;

}

public Integer getId() { return id; }

public void setId(Integer id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public Double getSalary() { return salary; }

public void setSalary(Double salary) { this.salary = salary; }

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + "]";

}

}

* ***EmployeeRepository.java:***

package com.cognizant.Employee;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

* ***EmployeeService.java:***

package com.cognizant.Employee;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public List<Employee> getAllEmployees() {

return employeeRepository.findAll();

}

@Transactional

public Employee getEmployeeById(Integer id) {

return employeeRepository.findById(id).orElse(null);

}

@Transactional

public Employee saveEmployee(Employee employee) {

return employeeRepository.save(employee);

}

@Transactional

public void deleteEmployee(Integer id) {

employeeRepository.deleteById(id);

}

}

* ***EmployeeApplication.java:***

package com.cognizant.Employee;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import java.util.List;

@SpringBootApplication

public class EmployeeApplication {

private static EmployeeService employeeService;

public static void main(String[] args) {

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

employeeService = context.getBean(EmployeeService.class);

System.out.println("=== All Employees ===");

testGetAllEmployees();

}

private static void testGetAllEmployees() {

List<Employee> employees = employeeService.getAllEmployees();

for (Employee emp : employees) {

System.out.println(emp);

}

}

}

Output:

