1.CONFIGURING A BASIC SPRING APPLICATION

Main.java

package ict.spring;

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");

Service service = (Service) context.getBean("service");

service.addBook("The Great Gatsby");

((ClassPathXmlApplicationContext) context).close();

}

}

Service.java

package ict.spring;

public class Service {

private Book book;

public void setBook(Book book) {

this.book = book;

}

public void addBook(String bookName) {

System.*out*.println("Adding book: " + bookName);

book.saveBook(bookName);

}

}

Book.java

package ict.spring;

public class Book {

public void saveBook(String bookName) {

System.*out*.println("Book saved to repository: " + bookName);

}

}

ApplicationContent.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="book" class="ict.spring.Book"/>

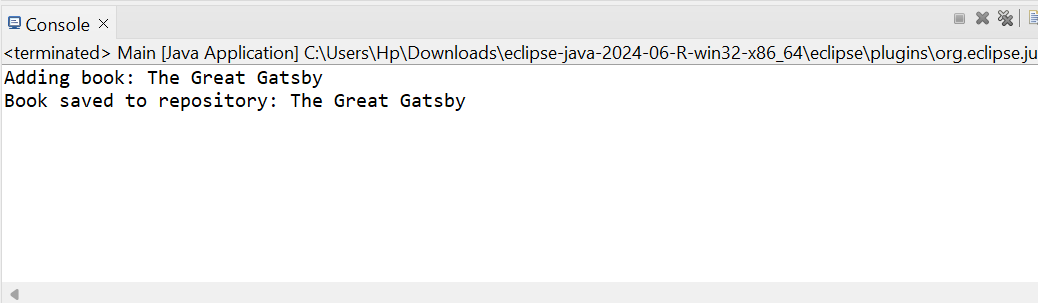
<bean id="service" class="ict.spring.Service">

<property name="book" ref="book"/>

</bean>

</beans>

OUTPUT:



2.IMPLEMENTING DEPENDENCY INJECTION

Main.java

package ict.spring;

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");

Service service = (Service) context.getBean("bookService");

service.addBook("The Great Gatsby");

((ClassPathXmlApplicationContext) context).close();

}

}

Service.java

package ict.spring;

public class Service {

private Book book;

public void setBook(Book book) {

this.book = book;

}

public void addBook(String bookName) {

System.*out*.println("Service: Adding book - " + bookName);

book.save(bookName);

}

}

Book.java

package ict.spring;

public class Book {

public void save(String bookName) {

System.*out*.println("Book: Saved - " + bookName);

}

}

ApplicationContent.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="book" class="ict.spring.Book" />

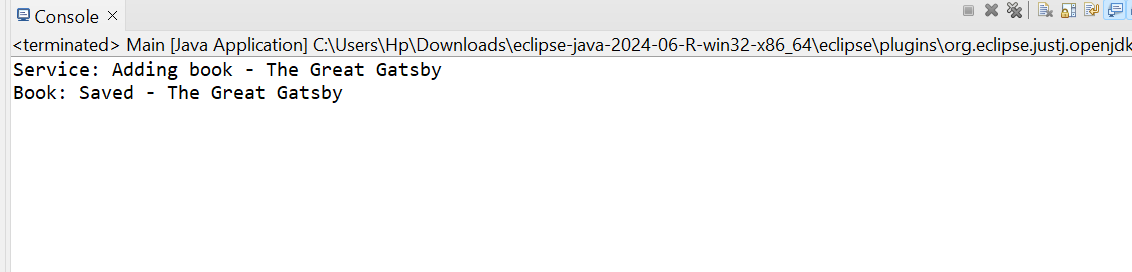
<bean id="bookService" class="ict.spring.Service">

<property name="book" ref="book" />

</bean>

</beans>

OUTPUT:



3.CREATING AND CONFIGURING A MAVEN PROJECT

Main.java

package ict.Library;

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 service = (BookService) context.getBean("bookService");

service.addBook("The Great Gatsby");

((ClassPathXmlApplicationContext) context).close();

}

}

BookService.java

package ict.Library;

public interface BookService {

void addBook(String bookName);

}

BookServiceImlp.java

package ict.Library;

public class BookServiceImpl implements BookService {

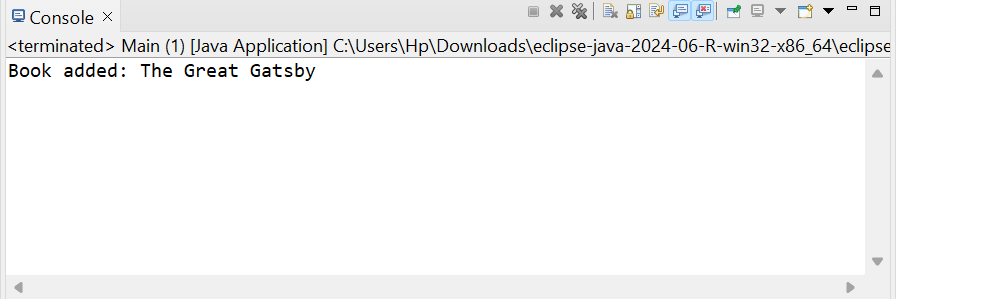
public void addBook(String bookName) {

System.*out*.println("Book added: " + bookName);

}

}

OUTPUT:



4. SPRING DATA JPA- QUICK EXAMPLE

OrmLearnApplication.java

package com.cognizant.orm\_learn;

import java.util.List;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.cognizant.orm\_learn.model.Country;

import com.cognizant.orm\_learn.service.CountryService;

@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);

*LOGGER*.info("Inside main");

*countryService* = context.getBean(CountryService.class);

*testGetAllCountries*();

}

private static void testGetAllCountries() {

*LOGGER*.info("Start");

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

for(Country country : countries) {

System.*out*.println(country);

}

*LOGGER*.info("End");

}

}

Country.java

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name="country")

public class Country {

@Id

@Column(name="code")

private String code;

@Column(name="name")

private String name;

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

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

}

}

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.orm\_learn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

CountryService.java

package com.cognizant.orm\_learn.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.orm\_learn.model.Country;

import com.cognizant.orm\_learn.repository.CountryRepository;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

Application.properties

spring.application.name=orm-learn

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

logging.level.org.hibernate.type.descriptor.sql=trace

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

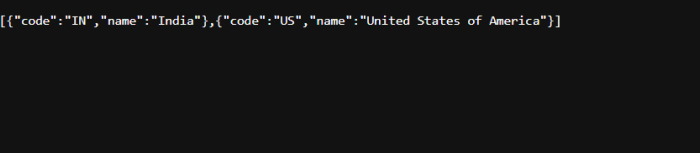
spring.datasource.username=root

spring.datasource.password=root@123

spring.jpa.hibernate.ddl-auto=validate

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

OUTPUT:



5. DIFFERENCE BETWEEN JPA, HIBERNATE AND SPRING DATA JPA

Employee.java

package com.example.demo.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Employee {

@Id

private int id;

private String name;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

Main.java

package com.example.demo;

import com.example.demo.model.Employee;

import jakarta.persistence.EntityManager;

import jakarta.persistence.EntityManagerFactory;

import jakarta.persistence.Persistence;

public class Main {

public static void main(String[] args) {

EntityManagerFactory emf = Persistence.createEntityManagerFactory("employee\_pu");

EntityManager em = emf.createEntityManager();

em.getTransaction().begin();

Employee e = new Employee();

e.setId(1);

e.setName("JPA Employee");

em.persist(e);

em.getTransaction().commit();

em.close();

emf.close();

}

}

Persistence.xml

<persistence xmlns="https://jakarta.ee/xml/ns/persistence"

version="3.0">

<persistence-unit name="employee\_pu">

<class>com.example.demo.model.Employee</class>

<properties>

<property name="jakarta.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/testdb"/>

<property name="jakarta.persistence.jdbc.user" value="root"/>

<property name="jakarta.persistence.jdbc.password" value="root"/>

<property name="jakarta.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>

<property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect"/>

<property name="hibernate.hbm2ddl.auto" value="update"/>

<property name="hibernate.show\_sql" value="true"/>

</properties>

</persistence-unit>

</persistence>

HibernateMain.java

package com.example.demo;

import com.example.demo.model.Employee;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

public class HibernateMain {

public static void main(String[] args) {

SessionFactory factory = new Configuration()

.configure("hibernate.cfg.xml")

.addAnnotatedClass(Employee.class)

.buildSessionFactory();

Session session = factory.getCurrentSession();

session.beginTransaction();

Employee e = new Employee();

e.setId(2);

e.setName("Native Hibernate Employee");

session.save(e);

session.getTransaction().commit();

factory.close();

}

}

EmployeeRepository.java

package com.example.demo.repository;

import com.example.demo.model.Employee;

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

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

DemoApplication.java

package com.example.demo;

import com.example.demo.model.Employee;

import com.example.demo.repository.EmployeeRepository;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DemoApplication implements CommandLineRunner {

@Autowired

private EmployeeRepository repo;

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

Employee e = new Employee();

e.setId(3);

e.setName("Spring Data JPA Employee");

repo.save(e);

}

}

Application.properties

spring.datasource.url=jdbc:mysql://localhost:3306/testdb

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

OUTPUT:

