Spring Data JPA - Hands On

# Hands on 1: Spring Data JPA - Quick Example

Software Pre-requisites:  
- MySQL Server 8.0  
- MySQL Workbench 8  
- Eclipse IDE for Enterprise Java Developers 2019-03 R  
- Maven 3.6.2  
  
Steps:  
- Go to https://start.spring.io/  
- Group: com.cognizant  
- Artifact Id: orm-learn  
- Description: Demo project for Spring Data JPA and Hibernate  
- Select: Spring Boot DevTools, Spring Data JPA, MySQL Driver  
- Generate, download, extract, and import in Eclipse.  
- Create schema in MySQL: create schema ormlearn;  
- application.properties:

# Logging  
logging.level.org.springframework=info  
logging.level.com.cognizant=debug  
logging.level.org.hibernate.SQL=trace  
logging.level.org.hibernate.type.descriptor.sql=trace  
logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n  
  
# DB Config  
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  
  
spring.jpa.hibernate.ddl-auto=validate  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

Build command:  
mvn clean package -Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050 -Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050 -Dhttp.proxyUser=123456  
  
Main class logging:  
private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);  
public static void main(String[] args) {  
 SpringApplication.run(OrmLearnApplication.class, args);  
 LOGGER.info("Inside main");  
}

# Hands on 4: Difference between JPA, Hibernate and Spring Data JPA

JPA: Specification only, no implementation.  
Hibernate: ORM implementation of JPA.  
Spring Data JPA: Abstraction over JPA + Hibernate, reduces boilerplate code.  
  
Hibernate Example:  
public Integer addEmployee(Employee employee){  
 Session session = factory.openSession();  
 Transaction tx = null;  
 Integer employeeID = null;  
 try {  
 tx = session.beginTransaction();  
 employeeID = (Integer) session.save(employee);  
 tx.commit();  
 } catch (HibernateException e) {  
 if (tx != null) tx.rollback();  
 e.printStackTrace();  
 } finally {  
 session.close();  
 }  
 return employeeID;  
}  
  
Spring Data JPA Example:  
@Repository  
public interface EmployeeRepository extends JpaRepository<Employee, Integer> { }  
  
@Service  
@Autowired  
private EmployeeRepository employeeRepository;  
  
@Transactional  
public void addEmployee(Employee employee) {  
 employeeRepository.save(employee);  
}

# Hands on 5 to 7: Country Service Implementation

CountryNotFoundException.java:  
public class CountryNotFoundException extends Exception { }  
  
CountryService.java:  
@Transactional  
public Country findCountryByCode(String code) throws CountryNotFoundException {  
 Optional<Country> result = countryRepository.findById(code);  
 if (!result.isPresent()) throw new CountryNotFoundException();  
 return result.get();  
}  
  
@Transactional  
public void addCountry(Country country) {  
 countryRepository.save(country);  
}  
  
OrmLearnApplication.java:  
private static void testGetAllCountries() {  
 LOGGER.info("Start");  
 List<Country> countries = countryService.getAllCountries();  
 LOGGER.debug("countries={}", countries);  
 LOGGER.info("End");  
}  
  
private static void getAllCountriesTest() {  
 LOGGER.info("Start");  
 Country country = countryService.findCountryByCode("IN");  
 LOGGER.debug("Country:{}", country);  
 LOGGER.info("End");  
}  
  
private static void testAddCountry() {  
 LOGGER.info("Start");  
 Country country = new Country();  
 country.setCode("ZZ");  
 country.setName("Newland");  
 countryService.addCountry(country);  
 LOGGER.info("End");  
}  
  
ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);  
countryService = context.getBean(CountryService.class);  
testGetAllCountries();  
getAllCountriesTest();  
testAddCountry();