

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра «Программная инженерия»

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №4

по предмету “Серверное программирование”

Настройка связи между java приложением и
базой данных. ORM Hibernate.

тема

Преподаватель

А.А. Даничев

подпись, дата

инициалы, фамилия

Студент ЗКИ21-16БВВ 031625881

08.04.2025

Е.М.Хорошко

подпись, дата

инициалы, фамилия

Красноярск, 2025

1. Задание.

В рамках данной практической работы необходимо настроить связь между java приложением и базой данных, используя ORM прослойку Hibernate. Также необходимо протестировать данную связь путём выполнения стандартных запросов.

2. Ход работы.

1. Создан новый Maven проект в Eclipse IDE:

GroupId: edu.sfu.

ArtifactId: lab4.

Структура проекта Рис.1

Файл pom.xml содержит информацию о зависимостях. (Приложение 1)

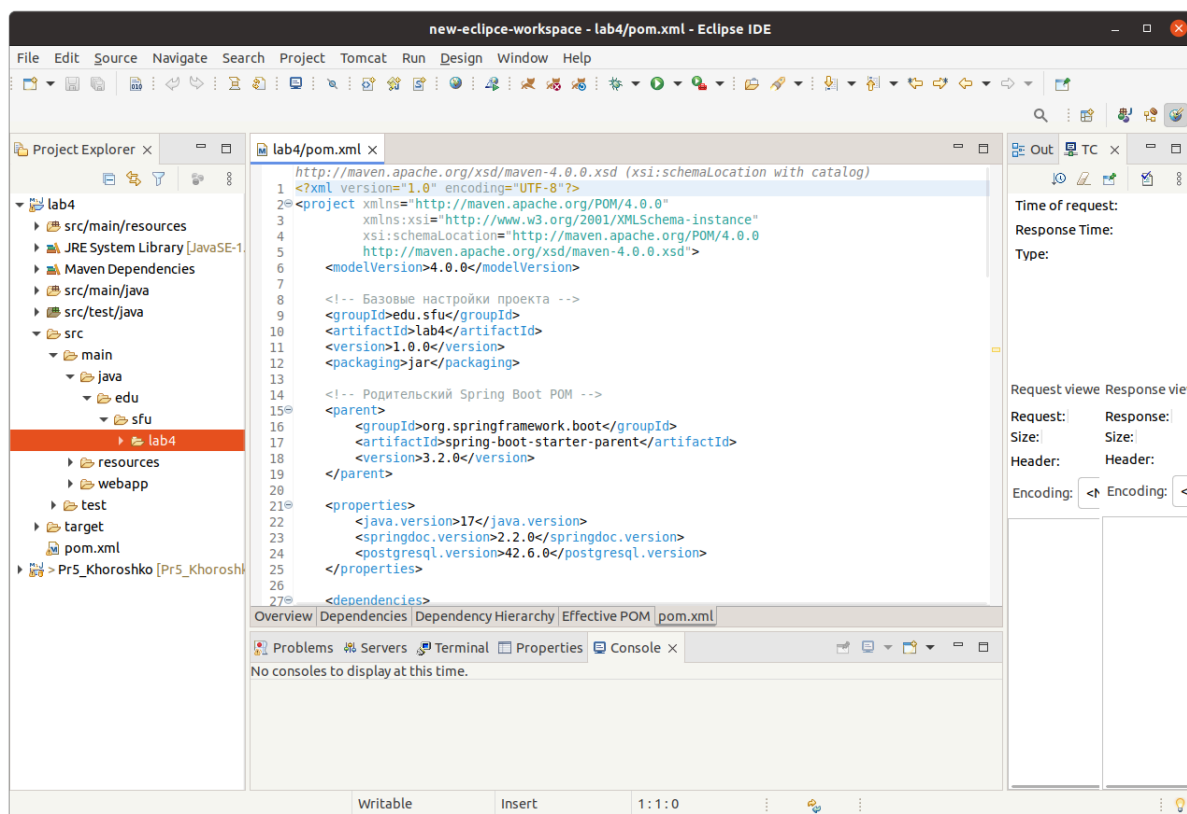


Рис.1 Структура проекта

2. В resources создан файл hibernate.cfg.xml. Данный файл содержит все настройки подключения к БД, а также mapping классов. (Приложение 1)

3. Создан java класс manager.DAO, для описания универсального взаимодействия с БД. В данном классе реализованы следующие статические поля:

```
ThreadLocal<Session> session = new ThreadLocal<Session>();
```

```
SessionFactory sessionFactory = new Configuration()
```

```
    .configure("hibernate.cfg.xml")
```

```
    .buildSessionFactory();
```

А также статические методы:

- getSession() - для получения новой сессии из sessionFactory. Данный метод проверяет, существует ли открытая сессия в DAO, и в случае, если ее нет – запрашивает новую сессию, помещает ее в статическое поле session и возвращает, как результат выполнения метода.
- begin() - получает сессию и начинает транзакцию (beginTransaction()).
- commit() - подтверждает ранее открытую транзакцию у текущей сессии (getTransaction().commit()).

4. Создан новый класс TestSrvs в пакете services. Данный класс является наследником DAO. В данном классе реализованы три тестовых метода (Приложение 1):

- getName - для получения имени страны по ID;
- getNamesInRange - для получения списка названий стран в диапазоне ID;
- createCountry - для создания новой записи в таблице Country. Метод обернут в пару begin() commit() для того чтобы результат сохранился, а так же вместо метода list использован метод executeUpdate.

5. Создан класс Main. В данном классе осуществлен вызов методов getName, getNamesInRange, createCountry (Приложение 1). Результат выводится в консоль через функцию System.out.println (Рис. 2). В таблицу Country вносится новая запись (Рис. 3)

```
kate@kate-IdeaPad-3-15ALC6: ~/SFU/7 семестр/Серверное программирование/практические задания/Lab4/lab4/target$ java -jar lab4-1.0.0.jar
00:24:42.876 [main] INFO org.hibernate.Version -- HHH0000412: Hibernate ORM core version 6.3.1.Final
00:24:43.631 [main] INFO org.hibernate.cache.internal.RegionFactoryInitiator -- HHH0000026: Second-level cache disabled
00:24:43.794 [main] WARN org.hibernate.orm.connections.pooling -- HHH10001002: Using built-in connection pool (not intended for production use)
00:24:43.798 [main] INFO org.hibernate.orm.connections.pooling -- HHH10001005: Loaded JDBC driver class: org.postgresql.Driver
00:24:43.799 [main] INFO org.hibernate.orm.connections.pooling -- HHH10001012: Connecting with JDBC URL [jdbc:postgresql://localhost:5432/jewelry_dbg]
00:24:43.799 [main] INFO org.hibernate.orm.connections.pooling -- HHH10001001: Connection properties: [password=****, user=postgres]
00:24:43.799 [main] INFO org.hibernate.orm.connections.pooling -- HHH10001003: Autocommit mode: false
00:24:43.802 [main] INFO org.hibernate.orm.connections.pooling -- HHH10001115: Connection pool size: 20 (min=1)
00:24:45.539 [main] WARN org.hibernate.orm.deprecation -- HHH90000025: PostgreSQLDialect does not need to be specified explicitly using 'hibernate.dialect'
(remove the property setting and it will be selected by default)
00:24:46.745 [main] INFO org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator -- HHH000489: No JTA platform available (set 'hibernat
e.transaction.jta.platform' to enable JTA platform integration)
Hibernate:
select
  c1_0.name
from
  country c1_0
where
  c1_0.id=?
Country name: Россия
Hibernate:
select
  c1_0.name
from
  country c1_0
where
  c1_0.id between ? and ?
Names in range: [Россия, Италия, Франция]
Hibernate:
INSERT
INTO
  Country
  (name, code)
VALUES
  (?, ?)
Created rows: 1
kate@kate-IdeaPad-3-15ALC6: ~/SFU/7 семестр/Серверное программирование/практические задания/Lab4/lab4/target$
```

Рис. 2 Результат выполнения программы

```
kate@kate-IdeaPad-3-15ALC6: ~/SFU/7 семестр/Сер... jewelry_dbg=# SELECT * FROM Country;
id |      name      | code
---+-----+---
 1 | Россия         | RU
 2 | Италия         | IT
 3 | Франция        | FR
 4 | США            | US
 5 | Китай          | CN
 6 | Германия       | DE
 7 | Великобритания | GB
 8 | Япония         | JP
 9 | Швейцария      | CH
10 | Canada         | CA
(10 rows)
```

Рис.3 Внесение новой записи в таблицу Country

Листинг кода pom.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>
  <!-- Базовые настройки проекта -->
  <groupId>edu.sfu</groupId>
  <artifactId>lab4</artifactId>
  <version>1.0.0</version>
  <packaging>jar</packaging>
  <!-- Родительский Spring Boot POM -->
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.2.0</version>
  </parent>
  <properties>
    <java.version>17</java.version>
    <springdoc.version>2.2.0</springdoc.version>
    <postgresql.version>42.6.0</postgresql.version>
  </properties>
  <dependencies>
    <!-- Spring Boot Web -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <!-- Spring Data JPA + Hibernate -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <!-- PostgreSQL Driver -->
    <dependency>
      <groupId>org.postgresql</groupId>
      <artifactId>postgresql</artifactId>
      <version>${postgresql.version}</version>
      <scope>runtime</scope>
    </dependency>
    <!-- Lombok -->
```

```

<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>>true</optional>
</dependency>
<!-- OpenAPI документация -->
<dependency>
  <groupId>org.springdoc</groupId>
  <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
  <version>${springdoc.version}</version>
</dependency>
<!-- Тестирование -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
</dependencies>
<build>
  <plugins>
    <!-- Maven компилятор -->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>${java.version}</source>
        <target>${java.version}</target>
      </configuration>
    </plugin>
    <!-- Spring Boot Maven Plugin -->
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <configuration>
        <mainClass>edu.sfu.lab4.Main</mainClass>
        <excludes>
          <exclude>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
          </exclude>
        </excludes>
      </configuration>
      <executions>
        <execution>
          <goals>

```

```

        <goal>repackage</goal>
    </goals>
</execution>
</executions>
</plugin>
</plugins>
</build>
</project>

```

Листинг кода hibernate.cfg.xml:

```

<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <!-- Database connection -->
        <property
name="hibernate.connection.driver_class">org.postgresql.Driver</property>
        <property
name="hibernate.connection.url">jdbc:postgresql://localhost:5432/jewelry_dbg
        </property>
        <property name="hibernate.connection.username">postgres</property>
        <property name="hibernate.connection.password">postgres</property>
        <property
name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.format_sql">true</property>
        <!-- Entity mappings -->
        <mapping class="edu.sfu.lab4.model.Country"/>
    </session-factory>
</hibernate-configuration>

```

Листинг кода DAO.java:

```

package edu.sfu.lab4.manager;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class DAO {
    protected static ThreadLocal<Session> session = new ThreadLocal<>();
}

```

```

protected static SessionFactory sessionFactory = new Configuration()
    .configure("hibernate.cfg.xml")
    .buildSessionFactory();

public static Session getSession() {
    Session currentSession = session.get();
    if (currentSession == null) {
        currentSession = sessionFactory.openSession();
        session.set(currentSession);
    }
    return currentSession;
}

public static void begin() {
    getSession().beginTransaction();
}

public static void commit() {
    getSession().getTransaction().commit();
}
}

```

Листинг кода TestSrvs.java:

```

package edu.sfu.lab4.services;

import edu.sfu.lab4.manager.DAO;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.query.Query;

public class TestSrvs extends DAO {
    // Получение имени страны по ID
    public static String getName(int id) {
        Session session = getSession();
        Query<String> query = session.createQuery(
            "SELECT c.name FROM Country c WHERE c.id = :id", String.class
        );
        query.setParameter("id", id);
        return query.uniqueResult();
    }

    // Получение списка имен в диапазоне ID
    public static List<String> getNamesInRange(int startId, int endId) {
        Session session = getSession();
        Query<String> query = session.createQuery(
            "SELECT c.name FROM Country c WHERE c.id BETWEEN :start AND :end",
            String.class
        );
    }
}

```



```

    );
    query.setParameter("start", startId);
    query.setParameter("end", endId);
    return query.list();
}
// Создание новой записи
public static int createCountry(String name, String code) {
    begin();
    Session session = getSession();
    Query<?> query = session.createNativeQuery(
        "INSERT INTO Country (name, code) VALUES (:name, :code)"
    );
    query.setParameter("name", name);
    query.setParameter("code", code);
    int result = query.executeUpdate();
    commit();
    return result;
}
}

```

Листинг кода Main.java:

```

package edu.sfu.lab4;
import edu.sfu.lab4.services.TestSrvs;
public class Main {
    public static void main(String[] args) {
        // Тест получения имени
        String countryName = TestSrvs.getName(1);
        System.out.println("Country name: " + countryName);
        // Тест получения диапазона
        System.out.println("Names in range: " + TestSrvs.getNamesInRange(1, 3));
        // Тест создания записи
        int created = TestSrvs.createCountry("Canada", "CA");
        System.out.println("Created rows: " + created);
    }
}

```