Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра экономической информатики

Лабораторная работа №8

**Разработка приложений на базе технологии Spring Boot**

Выполнил: Проверил:

Студент: гр. 714301 Кунцевич А.А.

Минск 2020

**Цель –** разработка консольного приложения на базе технологии Spring Boot для управления базой данных.

**Задание:** разработать приложение на базе Spring Boot, позволяющее управлять получением и сохранением в БД данных в форматах XML и JSON для проведения CRUD операций.

**Текст программы:**

package inpointapp.app;

import inpointapp.xml.JsonMarsh;

import inpointapp.xml.Marsh;

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

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

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import inpointapp.xml.XmlMarsh;

import java.util.Scanner;

@Component

public class Main {

public static void main(String[] args) {

ApplicationContext ctx = new AnnotationConfigApplicationContext("inpointapp");

Main app = ctx.getBean(Main.class);

app.start(ctx);

}

private Marsh marshaller;

public void start (ApplicationContext context){

while (true) {

System.out.println("Введите(без ковычек) <получить>, <послать>, либо <выйти>: ");

String action = cinString();

if (action.equals("получить")) {

writeFileName(context, true);

} else if (action.equals("послать")) {

writeFileName(context, false);

} else if (action.equals("выйти")) {

break;

} else {

System.out.println("Ошибка. Некорректный ответ!");

}

}

}

private void writeFileName(ApplicationContext context, boolean isFromDB){

System.out.println("Введите название файл для сохранения\\получения: ");

String fileName = "C:\\Users\\Атаджан\\okuw\\Laby\\3к-2\\РИС\\RisLab8\\" + cinString();

if(fileName.substring(fileName.length()-4).equals(".xml")) {

addMarshaller(fileName, context, XmlMarsh.class, isFromDB);

} else if(fileName.substring(fileName.length()-5).equals(".json")) {

addMarshaller(fileName, context, JsonMarsh.class, isFromDB);

} else {

System.out.println("Ошибка. Неверный формат!");

}

}

private void addMarshaller(String fileName, ApplicationContext context, Class<? extends Marsh> className,

boolean isFromDB){

marshaller = context.getBean(className);

if(isFromDB) marshaller.doMarsh(fileName);

else marshaller.doUnMarsh(fileName);

}

private static String cinString(){

Scanner cin = new Scanner(System.in);

if(cin.hasNext()){

return String.valueOf(cin.nextLine());

} else {

System.out.println("Вы ввели: '" + cin.next() + "'. Это не число. Попробуйте заново...");

return cinString();

}

}

}

package inpointapp.app;

import inpointapp.xml.Databaser;

import inpointapp.xml.JsonMarsh;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import inpointapp.xml.XmlMarsh;

public class Config {

@Configuration

public class ConfigurationBean {

@Bean

public XmlMarsh XmlMarsh(){

return new XmlMarsh();

}

@Bean

public JsonMarsh JsonMarsh(){

return new JsonMarsh();

}

@Bean

public Databaser Databaser(){

return new Databaser();

}

}

}

package inpointapp.xml;

import inpointapp.generated.Clients;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import org.hibernate.cfg.Configuration;

import org.springframework.stereotype.Repository;

import java.util.List;

@Repository("Databaser")

public class Databaser {

public SessionFactory getSessionFactory(){

SessionFactory sessionFactory = null;

try {

Configuration configuration = new Configuration().configure();

configuration.addAnnotatedClass(Clients.Client.class);

StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder().applySettings(configuration.getProperties());

sessionFactory = configuration.buildSessionFactory(builder.build());

} catch (Exception e) {

e.printStackTrace();

System.out.println("Исключение!" + e);

} finally {

return sessionFactory;

}

}

public final void addClient(final Clients.Client client) {

Session s = getSessionFactory().openSession();

s.getTransaction().begin();

s.persist(client);

s.getTransaction().commit();

}

public final List<Clients.Client> getClients() {

return getSessionFactory().openSession().createCriteria(Clients.Client.class).list();

}

public final void clearAll() {

Session sessions = getSessionFactory().openSession();

List<Clients.Client> clients = sessions.createCriteria(Clients.Client.class).list();

sessions.getTransaction().begin();

for (int i = 0; i < clients.size() ; i++) {

sessions.delete(clients.get(i));

}

sessions.getTransaction().commit();

}

}

package inpointapp.xml;

import com.google.gson.Gson;

import com.google.gson.GsonBuilder;

import inpointapp.generated.Clients;

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

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

import org.springframework.stereotype.Service;

import java.io.\*;

import java.util.List;

@Service

@Qualifier("JsonMarsh")

public class JsonMarsh implements Marsh{

private static GsonBuilder builder = new GsonBuilder();

private static Gson gson = builder.create();

List<Clients.Client> clients;

@Autowired

@Qualifier("Databaser")

private Databaser databaser;

public void doMarsh(String fileName) {

try (Writer writer = new FileWriter(fileName)){

clients = databaser.getClients();

gson.toJson(new Clients(clients), writer);

gson.toJson(clients);

printResult(fileName, "Записаны данные из БД в JSON Файл" );

} catch (Exception e) {

System.out.println("Ошибка сохранения данных в JSON: \"" + e.getMessage() + "\"");

}

}

public void doUnMarsh(String fileName) {

try (Reader reader = new FileReader(fileName)){

clients = (gson.fromJson(reader, Clients.class)).getClient();

builder.setPrettyPrinting().create().toJson(clients);

if(clients.size() > 0) databaser.clearAll();

for (int i = 0; i < clients.size(); i++) {

databaser.addClient(clients.get(i));

}

printResult(fileName, "Записаны данные в БД из JSON Файла" );

} catch (Exception e) {

System.out.println("Ошибка получения данных из JSON: \"" + e.getMessage() + "\"");

}

}

private void printResult(String fileName, String message){

System.out.println(message + " \"" + fileName + "\". Количество элементов: " + clients.size());

}

}

**Скриншоты работы:**









