

Лабораторная работа №7

Технология программирования

*«Организация в Java хранения данных в базе данных»*

Студент: Молнер В.С.

Группа: АВТ-814

Вариант: 4

Преподаватель: Михайленко Д.А.



НОВОСИБИРСК 2020

Оглавление

[Задание 1](#_Toc38223051)

[Ход работы 1](#_Toc38223052)

[Вывод 1](#_Toc38223053)

# Задание

1. Изучить особенности реализации хранения данных в базе данных в приложениях Java.
2. Доработать программу, созданную в лабораторной работе № 5:

* Добавить в меню пункты сохранение и загрузку объектов в базу данных. Можно использовать любую СУБД.
* Загрузку и сохранение объектов только заданного типа.

# Ход работы

Для выполнения задания была выбрана база данных MySQL.

В Workbench была создана таблица и добавлены 4 поля.

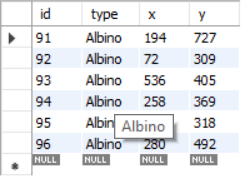


Рисунок - Пример таблицы

Для работы с базой данных был создан класс DBSaver. В него добавлены 6 публичных и 1 приватный метод.

import java.sql.\*;  
  
public class DBSaver {  
  
 private static final String *URL* = "jdbc:mysql://localhost:3306/mydbrabbits";  
 private static final String *USER* = "root";  
 private static final String *PASSWORD* = "2046";  
  
 public DBSaver() { }  
  
 public void SaveAll() {  
 DeleteAll();  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 PreparedStatement preparedStatement = connection.prepareStatement("insert into rabbit values(?,?,?,?)")) {  
 for (int i = 0; i < Singleton.*getVector*().size(); i++) {  
 preparedStatement.setInt(1,0);  
 if (Singleton.*getVector*().elementAt(i) instanceof Albino)  
 preparedStatement.setString(2,"Albino");  
 else  
 preparedStatement.setString(2,"Ordinary");  
 preparedStatement.setInt(3,Singleton.*getVector*().elementAt(i).getX());  
 preparedStatement.setInt(4,Singleton.*getVector*().elementAt(i).getY());  
  
 preparedStatement.execute();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void SaveAlbino() {  
 DeleteAll();  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 PreparedStatement preparedStatement = connection.prepareStatement("insert into rabbit values(?,?,?,?)")) {  
 for (int i = 0; i < Singleton.*getVector*().size(); i++) {  
 if (Singleton.*getVector*().elementAt(i) instanceof Albino) {  
 preparedStatement.setInt(1, 0);  
 preparedStatement.setString(2, "Albino");  
 preparedStatement.setInt(3, Singleton.*getVector*().elementAt(i).getX());  
 preparedStatement.setInt(4, Singleton.*getVector*().elementAt(i).getY());  
  
 preparedStatement.execute();  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void SaveOrdinary() {  
 DeleteAll();  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 PreparedStatement preparedStatement = connection.prepareStatement("insert into rabbit values(?,?,?,?)")) {  
 for (int i = 0; i < Singleton.*getVector*().size(); i++) {  
 if (Singleton.*getVector*().elementAt(i) instanceof Ordinary) {  
 preparedStatement.setInt(1, 0);  
 preparedStatement.setString(2, "Ordinary");  
 preparedStatement.setInt(3, Singleton.*getVector*().elementAt(i).getX());  
 preparedStatement.setInt(4, Singleton.*getVector*().elementAt(i).getY());  
  
 preparedStatement.execute();  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void LoadAll() {  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 Statement statement = connection.createStatement()) {  
 ResultSet resultSet = statement.executeQuery("select \* from rabbit");  
 while (resultSet.next()) {  
 if (resultSet.getString(2).equals("Albino"))  
 Singleton.*getVector*().add(new Albino(resultSet.getInt(3),resultSet.getInt(4),Habitat.*gui*.cumulativePeriod));  
 else  
 Singleton.*getVector*().add(new Ordinary(resultSet.getInt(3),resultSet.getInt(4),Habitat.*gui*.cumulativePeriod));  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void LoadAlbino() {  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 Statement statement = connection.createStatement()) {  
 ResultSet resultSet = statement.executeQuery("select \* from rabbit where type = 'Albino'");  
 while (resultSet.next()) {  
 Singleton.*getVector*().add(new Albino(resultSet.getInt(3),resultSet.getInt(4),Habitat.*gui*.cumulativePeriod));  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void LoadOrdinary() {  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 Statement statement = connection.createStatement()) {  
 ResultSet resultSet = statement.executeQuery("select \* from rabbit where type = 'Ordinary'");  
 while (resultSet.next()) {  
 Singleton.*getVector*().add(new Ordinary(resultSet.getInt(3),resultSet.getInt(4),Habitat.*gui*.cumulativePeriod));  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void DeleteAll() {  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);  
 Statement statement = connection.createStatement()) {  
 statement.execute("delete from rabbit");  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
}

Для взаимодействия с пользователем были добавлены соответствующие пункты в главное меню.

# Вывод

Были изучены несколько команд для работы с MySQL. Также была доработана предыдущая лабораторная работа.