|  |
| --- |
| https://lh6.googleusercontent.com/QcftzNtI05T0Y6fjdSh1Rr2rt8oqZ1IvnLvbn1jLJ7CCyteVir3k-xBLv4SL1wAgWJsRhmmJSR0UW-RP63_GQenE4vVWv05BRoZTsmIcBccVTnfxwmsnNMvjg599x9SqZd8E3dkd |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждениевысшего образования«МИРЭА - Российский технологический университет»РТУ МИРЭА |

Институт Информационных Технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения (ИППО)

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №9** | |
| **по дисциплине** | |
| «Разработка клиент-серверных приложений» | |
|  | |
| Выполнил студент группы ИКБО-12-18 | Копотов М.А. |
| Принял ассистент кафедры ИППО | Строганкова Н.В. |

|  |  |  |
| --- | --- | --- |
| Практическая работа выполнена | «\_\_»\_\_\_\_\_\_\_2020 г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2020 г. | *(подпись руководителя)* |

Москва 2020

**Содержание**

[1. Цель практической работы 3](#_Toc528748841)

[2. Задание 3](#_Toc528748842)

[3. Поэтапное выполнение практической работы 3](#_Toc528748843)

[5. Код программы 4](#_Toc528748844)

[6. Результат работы 4](#_Toc528748845)

[7. Вывод 4](#_Toc528748846)

[Список использованных источников 5](#_Toc528748847)

# 1 Цель практической работы

Знакомство с технологиями ORM и реализация взаимодействия с базой данных посредством ORM Hibernate.

# Задание

1. Разобраться с ORM-технологией и Hibernate библиотекой, а также с JDBC концепцией;
2. Опираясь на наработки предыдущей практической работы, реализовать подключение к базе данных с помощью библиотек JDBC (для каждой базы данных своя JDBC-библиотека), дальнейшее взаимодействие с базой данных должно осуществляться с помощью Hibernate. (Определение используемой базы данных на выбор студента, но желательно PostgreSQL):
3. Создание таблицы в базе данных: а) структура таблицы берется из предыдущей практической работы; б) создать класс, соответствующий структуре таблицы, с элементами аннотаций для взаимодействия с базой данных через библиотеку Hibernate;
4. Взаимодействие с таблицей необходимо осуществить на веб-странице, которая была создана ранее, и должна позволить выполнить следующие функции: удаления, обновления, добавления и отображения строк.

# Поэтапное выполнение практической работы

1. Все данные таблицы должны быть представлены в виде БД.

# Код программы

Server.java

package src;  
  
import java.io.IOException;  
import java.net.ServerSocket;  
  
public class Server {  
  
 public static void main(String[] args) {  
  
 try {  
 ServerSocket serverSocket = new ServerSocket(8080);  
 System.*out*.println("Сервер запущен на порту 8080!");  
 while (true) {  
 Client client = new Client(serverSocket.accept());  
  
 Thread thread = new Thread(client);  
 thread.start();  
 }  
  
 } catch (IOException e){  
 System.err.println(e);  
 }  
 }  
}

Client.java

package src;  
  
import java.io.\*;  
import java.net.Socket;  
import java.nio.file.Files;  
import java.nio.file.Paths;  
import java.util.Date;  
  
public class Client implements Runnable {  
  
 private Socket clientSocket;  
  
 public Client(Socket clientSocket) {  
 this.clientSocket = clientSocket;  
 }  
  
 public String getPage(String pageName) {  
 String filePath = "/Users/mikekopotov/Desktop/РКСП/PR8/src/public/" + pageName;  
 String pageData = "";  
 try {  
 pageData = Files.readString(Paths.get(filePath));  
 }catch (IOException e) {  
 e.printStackTrace();  
 }  
 return pageData;  
 }  
  
 @Override  
 public void run() {  
 //Инициализация  
 BufferedReader inStream;  
 PrintWriter headerPrintWriter;  
 BufferedWriter outDataStream;  
 try {  
 inStream = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));  
 String str = inStream.readLine();  
 System.out.println("-: " + str);  
 String request = str.split(" ")[1].substring(1);  
 System.out.println("Запрос: " + request);  
 //обрезать запрос типа '8080/about/'  
 if (!request.equals("") && !request.contains("?") && !request.contains("css") && !request.contains("js"))  
 request = request.substring(0, request.length() - 1);  
  
 String currentPage = "";  
 String contentType = "";  
 String restResponse = "";  
 boolean isRestUsed = false;  
  
 //Форматирование запроса  
 if(!request.contains("api")){  
 if(request.contains(".css")) {  
 request = request.substring(request.indexOf("css"));  
 currentPage = getPage(request);  
 contentType = "Content-type: text/css";  
 }  
 else if(request.contains(".js")){  
 request = request.substring(request.indexOf("js"));  
 currentPage = getPage(request);  
 contentType = "Content-type: text/javascript";  
 }  
 else {  
 switch (request) {  
 case "about":  
 currentPage = getPage("about.html");  
 break;  
 case "task":  
 currentPage = getPage("task.html");  
 break;  
 case "table":  
 currentPage = getPage("table.html");  
 break;  
 default:  
 currentPage = getPage("index.html");  
 break;  
 }  
 contentType = "Content-type: text/html; charset=utf-8";  
 }  
 }  
 else{  
 contentType = "Content-type: application/json; charset=utf-8";  
 Functions rest = new Functions(request);  
 restResponse = rest.getResponse();  
 isRestUsed = true;  
 }  
  
 //передача клиенту  
 headerPrintWriter = new PrintWriter(clientSocket.getOutputStream());  
 outDataStream = new BufferedWriter(new OutputStreamWriter(clientSocket.getOutputStream()));  
  
 headerPrintWriter.println("HTTP/1.1 200 OK");  
 headerPrintWriter.println("Server: Java HTTP Server : 1.0");  
 headerPrintWriter.println("Date: " + new Date());  
 headerPrintWriter.println(contentType);  
 headerPrintWriter.println("Accept-Language: \*");  
 headerPrintWriter.println();  
 headerPrintWriter.flush();  
  
 System.out.println("Итоговый запрос: " + request);  
  
 if(!isRestUsed){  
 System.out.println("Отправлен: файл");  
 outDataStream.write(currentPage);  
 outDataStream.flush();  
 }  
 else{  
 System.out.println("Отправлено: " + restResponse);  
 outDataStream.write(restResponse);  
 outDataStream.flush();  
 }  
  
 inStream.close();  
 headerPrintWriter.close();  
 outDataStream.close();  
 Thread.currentThread().interrupt();  
  
 } catch (IOException e) {  
 System.out.println(e.toString());  
 }  
 }  
}

Functions.java

package src;  
  
import java.io.Console;  
import java.io.UnsupportedEncodingException;  
import java.net.URLDecoder;  
import java.nio.charset.StandardCharsets;  
  
public class Functions {  
 private String query;  
 private String func;  
  
 private static RobotsTable *robotsTable* = new RobotsTable();  
  
 public Functions(String request) {  
 String query = request.substring(request.indexOf("?") + 1);  
 this.query = query;  
 if(query.contains("&")){  
 this.func = query.substring(query.indexOf("func") + 5, query.indexOf("&"));  
 }else {  
 this.func = query.substring(query.indexOf("func") + 5);  
 }  
 }  
  
 public String getVar(String varKey) {  
 String resultVar = "";  
 if(this.query.contains(varKey)) {  
 if (this.query.indexOf("&", this.query.indexOf(varKey)) == -1) {  
 resultVar = this.query.substring(this.query.indexOf(varKey) + varKey.length() + 1);  
 } else {  
 resultVar = this.query.substring(this.query.indexOf(varKey) + varKey.length() + 1, this.query.indexOf("&", this.query.indexOf(varKey)));  
 }  
 }  
 return resultVar;  
 }  
  
 public String getCalcResult() {  
 try {  
 int a = Integer.*parseInt*(getVar("calc-a"));  
 int b = Integer.parseInt(getVar("calc-b"));  
 String sigh = getVar("calc-s");  
 switch (sigh) {  
 case "+":  
 return String.valueOf(a + b);  
 case "\*":  
 return String.*valueOf*(a \* b);  
 case "/":  
 return String.*valueOf*(a / b);  
 case "-":  
 return String.*valueOf*(a - b);  
 default:  
 return "Неизвестный знак";  
 }  
  
 } catch (Exception e) {  
 return e.getMessage();  
 }  
 }  
  
 public String robotsFun(String queryUrl) {  
 String mode;  
  
 mode = getVar("mode");  
 String response = "";  
  
 int studentID = 0;  
 String name = "";  
 String desc = "";  
 String skill1 = "";  
 String skill2 = "";  
  
 if(this.query.contains("id")) {  
 studentID = Integer.*parseInt*(getVar("id"));  
 }  
 if(this.query.contains("name")) {  
 name = getVar("name");  
 }  
 if(this.query.contains("description")) {  
 desc = getVar("description");  
 }  
 if(this.query.contains("skill1")) {  
 skill1 = getVar("skill1");  
 }  
 if(this.query.contains("skill2")) {  
 skill2 = getVar("skill2");  
 }  
  
 switch (mode) {  
 case "get":  
 response = *robotsTable*.getRobots();  
 break;  
 case "delete":  
 response = *robotsTable*.deleteRobot(studentID);  
 break;  
 case "update":  
 response = *robotsTable*.updateRobot(  
 studentID,  
 name,  
 desc,  
 skill1,  
 skill2  
 );  
 break;  
 case "add":  
 response = *robotsTable*.addRobot(  
 name,  
 desc,  
 skill1,  
 skill2  
 );  
 break;  
 }  
 return response;  
 }  
  
 public String getResponse() {  
 try {  
 this.query = URLDecoder.*decode*(this.query, StandardCharsets.*UTF\_8*.name());  
 }  
 catch (UnsupportedEncodingException e){  
 e.printStackTrace();  
 }  
 String response = "";  
 if(this.func.equals("calc")){  
 response = getCalcResult();  
 }  
 else if(this.func.equals("table")){  
 response = robotsFun(this.query);  
 }  
 return response;  
 }  
}

Table.html

<!DOCTYPE html>  
<html lang="ru">  
 <head>  
 <meta charset="UTF-8">  
 <title>Таблица</title>  
 <link rel="stylesheet" href="css/style.css">  
 <script src="js/jquery.min.js"></script>  
 <script src="js/main.js"></script>  
 </head>  
 <body>  
 <header>  
 <a href="/about/">Обо мне</a>  
 <a href="/task/">Индивидуальное задание №13</a>  
 <a href="/table/">Таблица</a>   
 </header>   
 <section class="robot-table">  
 <div class="robot-table\_\_wrappper">  
 <div class="robot\_\_add">  
 <button class="button" id="add-robot">Добавить</button>  
 </div>  
 <div class="robot-table--data">  
 <div class="robot\_\_columns">  
 <div class="column column--delete">Удаление</div>  
 <div class="column column--update">Изменение</div>  
 <div class="column column--name">Название</div>  
 <div class="column column--lname">Описание</div>  
 <div class="column column--mname">Способность 1</div>  
 <div class="column column--yofb">Способность 2</div>  
 </div>  
 <div class="robot\_\_data">  
   
 </div>  
 </div>  
 </div>  
 </section>  
 </body>  
 <div class="popup" style="display: none;">  
 <div class="popup\_\_wrapper">  
 <h1 class="popup-func\_\_name"></h1>  
 <div class="popup\_\_methods" data-popup-func-type="" data-pupop-robot-id="">  
 <label for="robot-name">Введите название</label>  
 <input type="text" id="robot-name" value="Танкобот">  
 <label for="robot-description">Введите описание</label>  
 <input type="text" id="robot-description" value="Защитник">  
 <label for="robot-name">Введите способность 1</label>  
 <input type="text" id="robot-skill1" value="Стрелять">  
 <label for="robot-name">Введите способность 2</label>  
 <input type="text" id="robot-skill2" value="Защищать">  
 <div class="popup\_\_buttons">  
 <button class="button" id="popup-func">Добавить</button>  
 <button class="button" id="popup-close">Отмена</button>  
 </div>  
 </div>  
 </div>  
 </div>  
</html>

# 5 Результат работы

Изображение выглядит как текст, монитор, электроника, снимок экрана

Автоматически созданное описание

**Рисунок 1.Первоначальный вид БД при запуске**

**Изображение выглядит как текст, монитор, электроника, снимок экрана

Автоматически созданное описание**

**Рисунок 2. Удаление нижней строки**

**Изображение выглядит как текст, монитор, внутренний, электроника

Автоматически созданное описание**

**Рисунок 3. Обновление данных робота в третьей строке**

**Изображение выглядит как текст, монитор, электроника, снимок экрана

Автоматически созданное описание**

**Рисунок 4. Результат обновления последней строки БД**

# 6 Вывод

В данной практической работе мы ознакомились с технологиями ORM и реализация взаимодействия с базой данных посредством ORM Hibernate.

# Список использованных источников

1. https://tproger.ru/translations/java-intro-for-beginners/
2. https://www.sqlitetutorial.net/
3. https://www.jetbrains.com/ru-ru/idea/
4. https://developer.mozilla.org/ru/docs/Web/HTTP/Overview