

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Кафедра компьютерных технологий и программной инженерии

ОТЧЁТ
ЗАЩИЩЁН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель

должность, уч. степень, звание

подпись, дата

Путилова Н. В.

инициалы, фамилия

Отчёт по лабораторной работе №8

по курсу: Проектирование баз данных

СТУДЕНТ ГР. № 4932

номер группы

подпись, дата

С. И. Коваленко

инициалы, фамилия

Санкт-Петербург
2021

Цель работы: Проектирование взаимодействия базы данных и приложения

Обработка данных sql запроса в цикле

```
fun main(args: Array<String>) {
    JTableExamples()
}

class JTableExamples internal constructor() {
    var f: JFrame = JFrame()

    init {
        f.title = "DB1"

        val db = Jdbc()
        val data = db.readOwner()

        val columnNames = Vector<String>(4)
        columnNames.add("id")
        columnNames.add("Фамилия")
        columnNames.add("Имя")
        columnNames.add("Отчество")

        val j = JTable(DefaultTableModel(data, columnNames))
        j.setBounds(30, 40, 200, 300)

        val sp = JScrollPane(j)
        f.add(sp)
        f.setSize(500, 200)
        f.isVisible = true
    }
}

class Jdbc {
    /** JDBC Driver and database url */
    private val JDBC_DRIVER = "com.mysql.cj.jdbc.Driver"
    private val DATABASE_URL = "jdbc:mysql://localhost/bd_schema"

    private val USER = "admin"
    private val PASSWORD = "admin"

    private var statement : Statement
    private var connection : Connection

    init {
        println("Registering JDBC driver...")
        Class.forName(JDBC_DRIVER)
        println("Creating database connection...")
        connection = DriverManager.getConnection(DATABASE_URL, USER, PASSWORD)
        println("Executing statement...")
        statement = connection.createStatement()
    }

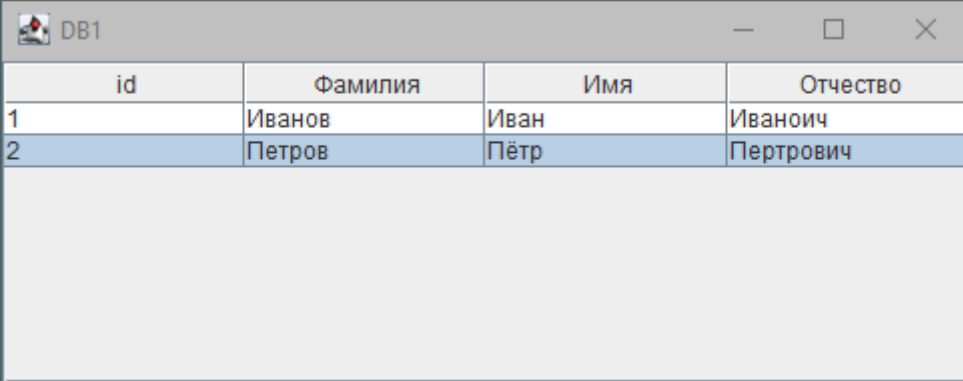
    fun finalize() {
        println("Closing connection and releasing resources...")
        statement.close()
        connection.close()
    }

    fun readOwner() : Vector<Vector<String>> {
        val sql = "SELECT * FROM owner"
        val resultSet = statement.executeQuery(sql)
        println("Retrieving data from database...")
    }
}
```

```

val ownerList = Vector<Vector<String>> ()
var i = 0
while (resultSet.next()) {
    val a = Vector<String>(4)
    a.add(resultSet.getInt("id_owner").toString())
    a.add(resultSet.getString("Surname"))
    a.add(resultSet.getString("Name"))
    a.add(resultSet.getString("Middle_name"))
    ownerList.add(a)
}
resultSet.close()
return ownerList;
}
}

```



id	Фамилия	Имя	Отчество
1	Иванов	Иван	Иванович
2	Петров	Пётр	Петрович

Обращение к базе данных с использованием DataGridView

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
namespace pbd88
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            MySqlConnectionStringBuilder mySql = new MySqlConnectionStringBuilder();
            mySql.Server = "127.0.0.1";
            mySql.Database = "db_schema";
            mySql.UserID = "admin";
            mySql.Password = "admin";
            DataTable table = new DataTable();
            string query = @"SELECT * from owner";
            using (MySqlConnection cnn = new MySqlConnection(mySql.ConnectionString))
            {
                cnn.ConnectionString = mySql.ConnectionString;
                MySqlCommand command = new MySqlCommand(query, cnn);
            }
        }
    }
}

```

```

        cnn.Open();
        using (MySqlDataReader dr = command.ExecuteReader())
        {
            if (dr.HasRows) table.Load(dr);
        }
        cnn.Close();
        dataGridView1.DataSource = table;
    }
}

```

```

namespace pbd8
{
    partial class Form1
    {
        private System.ComponentModel.IContainer components = null;

        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Код, автоматически созданный конструктором форм Windows
        private void InitializeComponent()
        {
            this.button1 = new System.Windows.Forms.Button();
            this.button2 = new System.Windows.Forms.Button();
            this.button3 = new System.Windows.Forms.Button();
            this.dataGridView1 = new System.Windows.Forms.DataGridView();
            this.listView1 = new System.Windows.Forms.ListView();
            this.listBox1 = new System.Windows.Forms.ListBox();
            ((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).BeginInit();
            this.SuspendLayout();
            //
            // button1
            //
            this.button1.Location = new System.Drawing.Point(140, 249);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(75, 23);
            this.button1.TabIndex = 0;
            this.button1.Text = "Показать";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click_1);
            //
            // dataGridView1
            //
            this.dataGridView1.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            this.dataGridView1.Location = new System.Drawing.Point(12, 12);
            this.dataGridView1.Name = "dataGridView1";
            this.dataGridView1.Size = new System.Drawing.Size(491, 231);
            this.dataGridView1.TabIndex = 3;
            //
            // listView1
            //
            /* this.listView1.HideSelection = false;
            this.listView1.Location = new System.Drawing.Point(500, 23);
            this.listView1.Name = "listView1";

```

```

this.listView1.Size = new System.Drawing.Size(242, 175);
this.listView1.TabIndex = 4;
this.listView1.UseCompatibleStateImageBehavior = false;
this.listView1.View = System.Windows.Forms.View.Details;*/
//
// Form1
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(800, 450);
//this.Controls.Add(this.listBox1);
//this.Controls.Add(this.listView1);
this.Controls.Add(this.dataGridView1);
//this.Controls.Add(this.button3);
//this.Controls.Add(this.button2);
this.Controls.Add(this.button1);
this.Name = "Form1";
this.Text = "Form1";
((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).EndInit();
this.ResumeLayout(false);

```

```

}

```

```

#endregion

```

```

private System.Windows.Forms.Button button1;
private System.Windows.Forms.DataGridView dataGridView1;
private System.Windows.Forms.ListView listView1;
private System.Windows.Forms.ListBox listBox1;

```

```

}
}

```

	id_owner	Sumame	Name	Middle_name
▶	1	Ivan	Ivanov	Ivanovich
	2	Petrov	Petr	Petrovich
*				

Показать

Обращение к базе данных с помощью Hibernate

Конфигурация Hibernate

```

<hibernate-configuration>
  <session-factory>

    <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property name="connection.url">jdbc:mysql://localhost:3306/bd_schema</property>
    <!-- jdbc:h2:file:/data/sample -->
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>

```

```

    <property name="connection.username">admin</property>
    <property name="connection.password">admin</property>
    <property name="hbm2ddl.auto">update</property>
    <mapping class="Owner"/>
</session-factory>
</hibernate-configuration>

```

Сущность для связи с таблицей

```

import javax.persistence.*

@Entity
@Table(name = "owner", schema = "bd_schema")
data class Owner(
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id_owner", nullable = false)
    var Id: Long? = null,

    @Column(name = "Name", nullable = false)
    var Name: String? = null,

    @Column(name = "Surname", nullable = false)
    var Surname: String? = null,

    @Column(name = "Middle_name", nullable = false)
    var Middle_name: String? = null
)

```

```

import java.util.*
import javax.swing.JFrame
import javax.swing.JScrollPane
import javax.swing.JTable
import javax.swing.table.DefaultTableModel

fun main(args: Array<String>) {
    JTableExamples()
}

class JTableExamples internal constructor() {
    var f = JFrame()
    var j: JTable

    init {
        f.title = "DB hibernate"

        val session = HibernateSessionFactory.createSession()
        val query = session.createQuery("from Owner")
        val result = query.resultList

        println(result.toString())
        val v = Vector<Vector<String>>()
        result.forEach { v.add(it.toVector()) }
        session.close()
        // Column Names
        val columnNames = Vector<String>(4)
        columnNames.add("id")
    }
}

```

```

columnNames.add("Фамилия")
columnNames.add("Имя")
columnNames.add("Отчество")
// Initializing the JTable
j = JTable(DefaultTableModel(v, columnNames))
j.setBounds(30, 40, 200, 300)
// adding it to JScrollPane
val sp = JScrollPane(j)
f.add(sp) // Frame Size
f.setSize(500, 200) // Frame Visible = true
f.isVisible = true
}
}

fun Any.toVector() : Vector<String>? {
    return if (this is Owner) {
        val vector = Vector<String>()
        vector.add(this.Id.toString())
        vector.add(this.Surnane)
        vector.add(this.Name)
        vector.add(this.Middle_name)
        vector
    } else {
        null
    }
}
}

```

DB hibernate			
id	Фамилия	Имя	Отчество
1	Иванов	Иван	Иваноич
2	Петров	Пётр	Пертрович