**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра ВТ**

отчет

**по лабораторной работе №2**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: Разработка интерфейса пользователя

|  |  |  |
| --- | --- | --- |
| Студентка гр. 2308 |  | Рымарь М.И. |
| Преподаватель |  | Павловский М.Г. |

Санкт-Петербург

2023

**Цель работы.**

Знакомство с правилами построения экранной формы.

**Задание (вариант 1).**

1. Из задания к курсовой работе выберите экранную форму, содержащую от 8 до 12 графических компонентов.
2. Опишите назначение экранной формы с указанием перечня вводимой и выводимой информации, а также списка функций, доступных пользователю.
3. С помощью стандартных средств рисования или с использованием демоверсии Balsamiq Mockups спроектируйте внешний вид экранной формы.
4. Для каждого нарисованного элемента экранной формы подберите из библиотеки java.awt или javax.swing подходящий графический компонент.
5. Выберите способ компоновки графических элементов на экранной форме.
6. Создайте класс «Приложение» и объявите в нем графические компоненты.
7. Разработайте метод построения и визуализации экранной формы, который создает и размещает объявленные графические компоненты с помощью выбранных классов компоновки. В код метода должны быть вставлены комментарии документации, отражающие процесс построения экранной формы.
8. Создайте и отобразите разработанную экранную форму.
9. Сгенерируйте документацию с помощью Javadoc и просмотрите ее в браузере.

**Выполнение работы.**

1. Описание макета экранной формы

Экранная форма отображает список видеокарт магазина электроники. Окно приложения имеет свойство изменения своего размера. Форма должна реализовывать функции сохранения списка видеокарт в файле и загрузки списка из файла. Также должны быть реализованы функции добавления видеокарты в список или удаления наименования из списка. Помимо этого, должна быть доступна функция перехода к карточке для редактирования данных о товаре и поиск товара по производителю или модели.

Макет экранной формы представлен на рисунке 1.

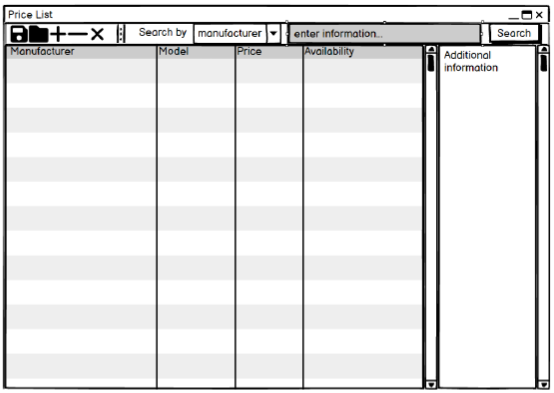
****

Рисунок 1 – Макет экранной формы

2. Разметка и содержимое экранной формы

Общий вид программы представлен на рисунке 2.

Панель инструментов JToolBar размещена менеджером граничного размещения BorderLayout .NORTH наверху экранной формы. На ней находится 5 кнопок JButton: сохранить файл, открыть файл, добавить, удалить, редактировать. Также в панели инструментов находится текстовое поле JLabel, выпадающий список JComboBox, текстовое поле JTextField, кнопка JButton.

Таблица JTable состоит из 4 столбцов – manufacturer, model, price, availability. Таблица находится в центре экранной формы.

Текстовое поле JTextArea находится в правой части экранной формы.

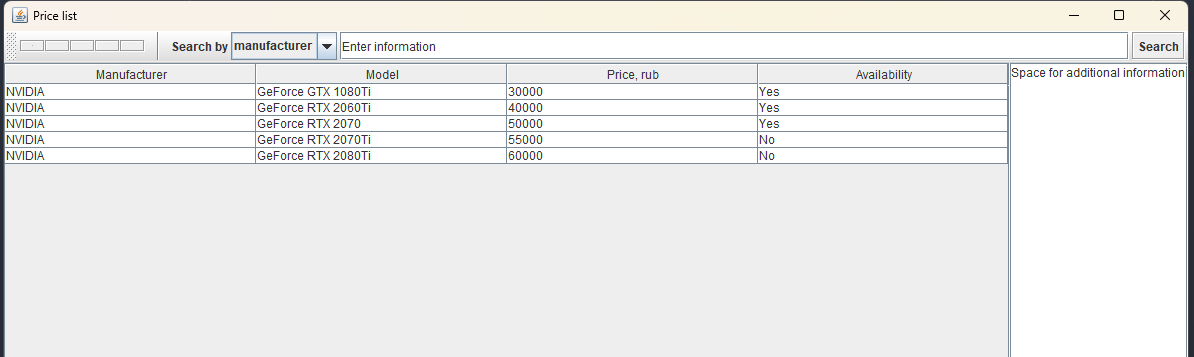


Рисунок 2 – Окно приложения

3. Текст документации, сгенерированный JavaDoc

Текст документации представлен в папке с исходным кодом программы. Часть документации из браузера представлена на рисунке 3.

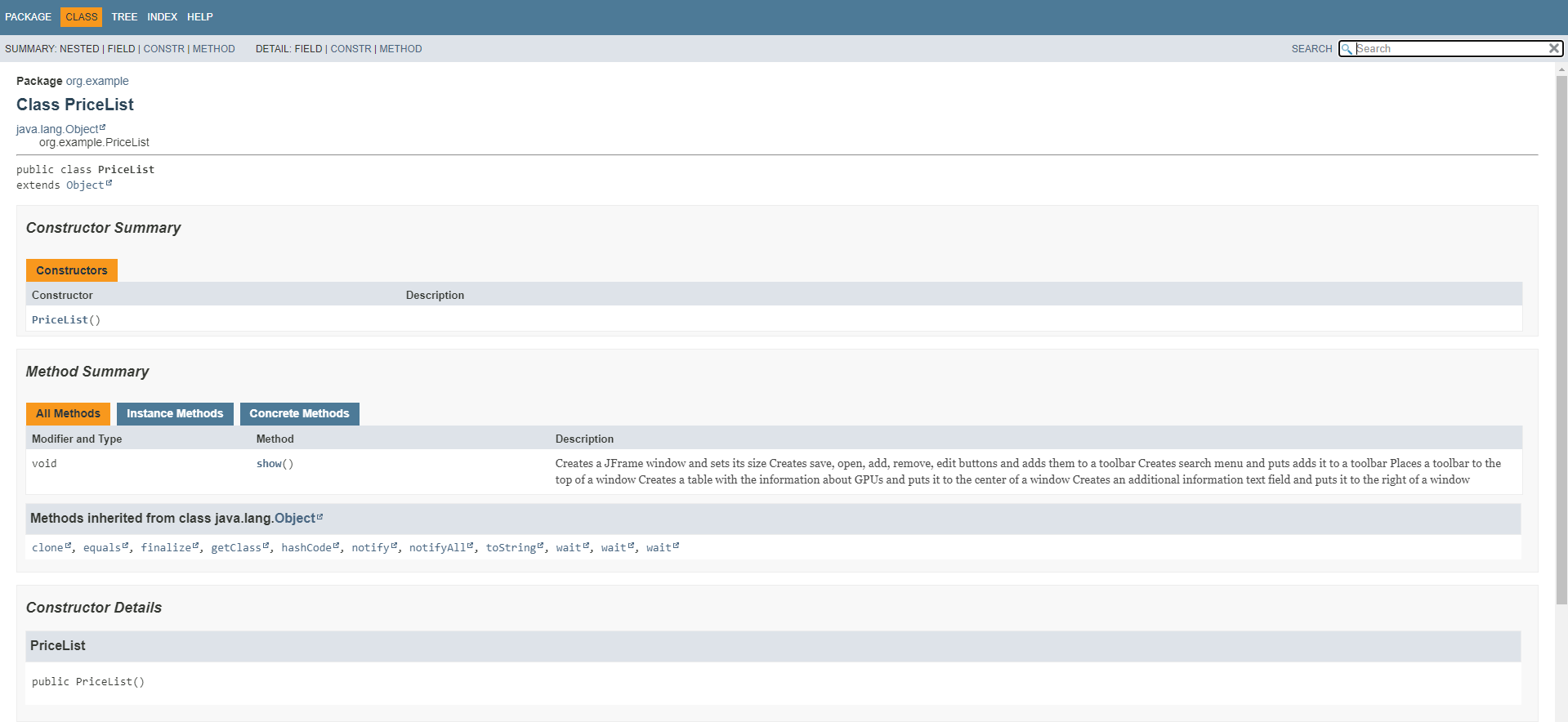


Рисунок 3 – Часть документации JavaDoc

**Выводы.**

В ходе выполнения лабораторной работы была освоена среда разработки IntelliJ IDEA. Были получены навыки в запуске и отладке приложения с графическим интерфейсом пользователя.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Файл Main.java:

package org.example;

public class Main {

// Creating and showing gui

public static void main(String[] args) {

new PriceList().show();

}

}

Файл PriceList.java:

package org.example;

/\* Подключение графических библиотек \*/

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

public class PriceList {

/\* Объявления графических компонентов \*/

private JFrame priceList;

private DefaultTableModel priceListModel;

private JButton save;

private JButton open;

private JButton add;

private JButton remove;

private JButton edit;

private JLabel searchByLabel;

private JComboBox searchBy;

private JTextField searchInformation;

private JButton searchButton;

private JToolBar toolBar;

private JTable GPUTable;

private JScrollPane tableScrollBar;

private JTextArea additionalInformation;

/\*\*

\* Creates a JFrame window and sets its size

\* Creates save, open, add, remove, edit buttons and adds them to a toolbar

\* Creates search menu and puts adds it to a toolbar

\* Places a toolbar to the top of a window

\* Creates a table with the information about GPUs and puts it to the center of a window

\* Creates an additional information text field and puts it to the right of a window

\*/

public void show() {

/\* Создание окна \*/

priceList = new JFrame("Price list");

priceList.setSize(1200, 600);

priceList.setLocation(300,300);

priceList.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

/\* Скейлинг изображений для правильного отображения в кнопках \*/

ImageIcon imageIconSave = new ImageIcon(new ImageIcon("src/main/resources/save.png").getImage().getScaledInstance(20, 20, Image.SCALE\_DEFAULT));

ImageIcon imageIconOpen = new ImageIcon(new ImageIcon("src/main/resources/open.png").getImage().getScaledInstance(20, 20, Image.SCALE\_DEFAULT));

ImageIcon imageIconAdd = new ImageIcon(new ImageIcon("src/main/resources/add.png").getImage().getScaledInstance(20, 20, Image.SCALE\_DEFAULT));

ImageIcon imageIconRemove = new ImageIcon(new ImageIcon("src/main/resources/remove.png").getImage().getScaledInstance(20, 20, Image.SCALE\_DEFAULT));

ImageIcon imageIconEdit = new ImageIcon(new ImageIcon("src/main/resources/edit.png").getImage().getScaledInstance(20, 20, Image.SCALE\_DEFAULT));

/\* Создание кнопок и прикрепление иконок \*/

save = new JButton(imageIconSave);

open = new JButton(imageIconOpen);

add = new JButton(imageIconAdd);

remove = new JButton(imageIconRemove);

edit = new JButton(imageIconEdit);

/\* Настройка подсказок для кнопок \*/

save.setToolTipText("Save");

open.setToolTipText("Open");

add.setToolTipText("Add");

remove.setToolTipText("Remove");

edit.setToolTipText("Edit");

/\* Настройка размера кнопок \*/

save.setPreferredSize(new Dimension(25,25));

open.setPreferredSize(new Dimension(25,25));

add.setPreferredSize(new Dimension(25,25));

remove.setPreferredSize(new Dimension(25,25));

edit.setPreferredSize(new Dimension(25,25));

/\* Создание поиска \*/

searchByLabel = new JLabel("Search by");

searchBy = new JComboBox(new String[]{"manufacturer", "model"});

searchInformation = new JTextField("Enter information");

searchButton = new JButton("Search");

/\* Добавление кнопок и поиска на панель инструментов \*/

toolBar = new JToolBar("Toolbar");

toolBar.add(save);

toolBar.add(open);

toolBar.add(add);

toolBar.add(remove);

toolBar.add(edit);

toolBar.add(Box.createHorizontalStrut(12));

toolBar.add(new JSeparator(SwingConstants.VERTICAL));

toolBar.add(Box.createHorizontalStrut(12));

toolBar.add(searchByLabel);

toolBar.add(Box.createHorizontalStrut(3));

toolBar.add(searchBy);

toolBar.add(Box.createHorizontalStrut(3));

toolBar.add(searchInformation);

toolBar.add(Box.createHorizontalStrut(3));

toolBar.add(searchButton);

/\* Размещение панели инструментов \*/

priceList.setLayout(new BorderLayout());

priceList.add(toolBar, BorderLayout.NORTH);

/\* Создание таблицы с данными \*/

String[] Columns = {"Manufacturer", "Model", "Price, rub", "Availability"};

String[][] Data = {{"NVIDIA", "GeForce GTX 1080Ti", "30000", "Yes"}, {"NVIDIA", "GeForce RTX 2060Ti", "40000", "Yes"},

{"NVIDIA", "GeForce RTX 2070", "50000", "Yes"}, {"NVIDIA","GeForce RTX 2070Ti", "55000", "No"},

{"NVIDIA","GeForce RTX 2080Ti", "60000", "No"}};

priceListModel = new DefaultTableModel(Data, Columns);

GPUTable = new JTable(priceListModel);

tableScrollBar = new JScrollPane(GPUTable);

/\* Размещение таблицы с данными \*/

priceList.add(tableScrollBar, BorderLayout.CENTER);

/\* Создание окна с доп. информацией \*/

additionalInformation = new JTextArea("Space for additional information");

/\* Размещение окна с доп. информацией \*/

priceList.add(additionalInformation, BorderLayout.EAST);

priceList.add(new JScrollPane(additionalInformation), BorderLayout.EAST);

priceList.setVisible(true);

}

}