Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №4**

**Дисциплина: Платформо-независимое программирование**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К. А. Корнилов

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В. И. Шиян

**Тема:** Библиотеки AWT и SWING для построения графического интерфейса пользователя. Обработка событий

**Цель**:

Освоить методы работы с графическим интерфейсом Java, ознакомиться с различными компоновщиками. Научиться обрабатывать события пользовательского интерфейса.

**Задание:**

1. Изменить текст метки в окне, написать код создания 3 меток и объяснить и положение в окне.
2. Написать программу, создающую 8-9 меток в окне. Объяснить их расположение при различных размерах окна.
3. Была написана программа, при запуске которой в главном окне появляется кнопка с счетчиком. Сделать, чтобы при значении счетчика 1 приложение закрывалось.
4. Написать программу, при запуске которой в главном окне появляются 2 кнопки, каждая из которых включает/отключает другую
5. Написать программу по заданию:



1. Составить диаграмму классов и последовательности для программы.

**Ход работы:**

1. Был изменен текст метки в окне и написан код, создающий 3 метки в окне.

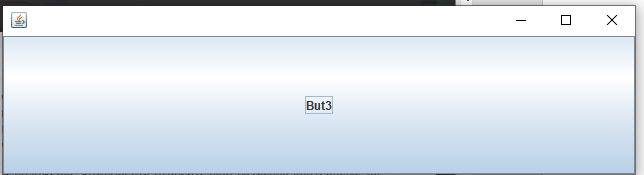


Рисунок 1 – Вид нового окна.

Созданные объекты будут занимать весь доступный размер окна. При этом объекты будут перекрывать друг друга, и пользователю будет виден только последний добавленный объект.

import javax.swing.\*;  
public class Main {  
 public static void main(String[] args) {  
 JFrame f = new JFrame();  
 f.setSize(300,200);  
 f.setLocation(500,200);  
  
 JButton b1 = new JButton("But1");  
 JButton b2 = new JButton("But2");  
 JButton b3 = new JButton("But3");  
 JLabel lab = new JLabel("Новая метка");  
 f.add(b1);  
 f.add(b2);  
 f.add(b3);  
 f.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 f.setVisible(true);  
 }  
}

1. Была написана программа, создающая 8 меток в окне с помощью элемента JPanel.

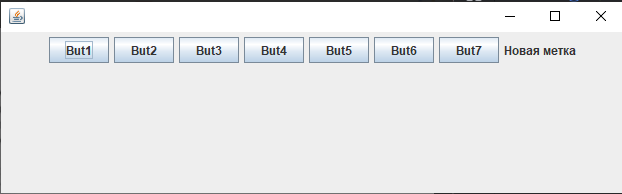


Рисунок 2 – Внешний вид окна.

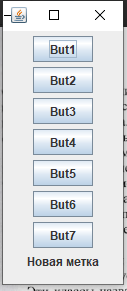


Рисунок 3 – Внешний вид окна минимального размера.

В данном случае объекты не будут занимать все окно. Также все объекты располагаются в 1 строку друг за другом в порядке их добавления в панель. Если для какого-то объекта не хватает места, то он переносится на следующую строку. Поэтому при минимальной ширине окна все объекты будут располагаться в 1 столбец.

1. Была написана программа, при запуске которой появляется кнопка с счетчиком, которая по достижению 1 выходит из программы.

Код программы:

import javax.swing.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
class Handler implements ActionListener{  
 JButton B;  
 public Handler(JButton bb){  
 B=bb;  
 }  
 @Override  
 public void actionPerformed(ActionEvent e){  
 String t = this.B.getText();  
 if(Integer.*parseInt*(t)>1){  
 this.B.setText(String.*valueOf*((Integer.*parseInt*(t)-1)));  
 }  
 else{  
 System.*exit*(0);  
 }  
 }  
}  
public class Windows4\_3 {  
 public Windows4\_3(){  
 JFrame f = new JFrame();  
 f.setSize(300,200);  
 f.setLocation(500,200);  
  
 JButton b1 = new JButton("10");  
 b1.addActionListener(new Handler(b1));  
  
 JPanel panel = new JPanel();  
 panel.add(b1);  
 f.add(panel);  
  
  
 f.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 f.setVisible(true);  
 }  
}

1. Была написана программа, при запуске которой в главном окне появляются две кнопки, каждая из которых включает/отключает другую.

Код программы:

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.MouseAdapter;  
import java.awt.event.MouseEvent;  
  
public class Windows4\_4 {  
 public Windows4\_4(){  
 JFrame frame = new JFrame("Окно верхнего уровня");  
  
 frame.setLayout(new FlowLayout());  
  
  
 JButton b1 = new JButton("Button 1");  
 JButton b2 = new JButton("Button 2");  
 b1.addMouseListener(new MouseAdapter() {  
 @Override  
 public void mouseClicked(MouseEvent e) {  
 super.mouseClicked(e);  
 if (b1.isEnabled()){  
 b2.setEnabled(!b2.isEnabled());  
 }  
  
 }  
 });  
  
 b2.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if(b2.isEnabled()){  
 b1.setEnabled(!b1.isEnabled());  
 }  
  
 }  
 });  
  
 frame.add(b1);  
 frame.add(b2);  
  
  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setSize(300,200);  
 frame.setVisible(true);  
 }  
}

EventQueue.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 Windows4\_4 a = new Windows4\_4();  
 }  
});

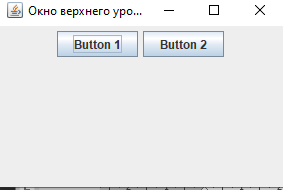


Рисунок 4 – Внешний вид окна.

1. Была написана программа, соответствующая следующему заданию:



Рисунок 5 – Задание.

Все объекты данной программы были преобразованы отдельные в классы. Из этих классов была построена иерархия классов. Для данной иерархии была построена uml диаграмма классов.

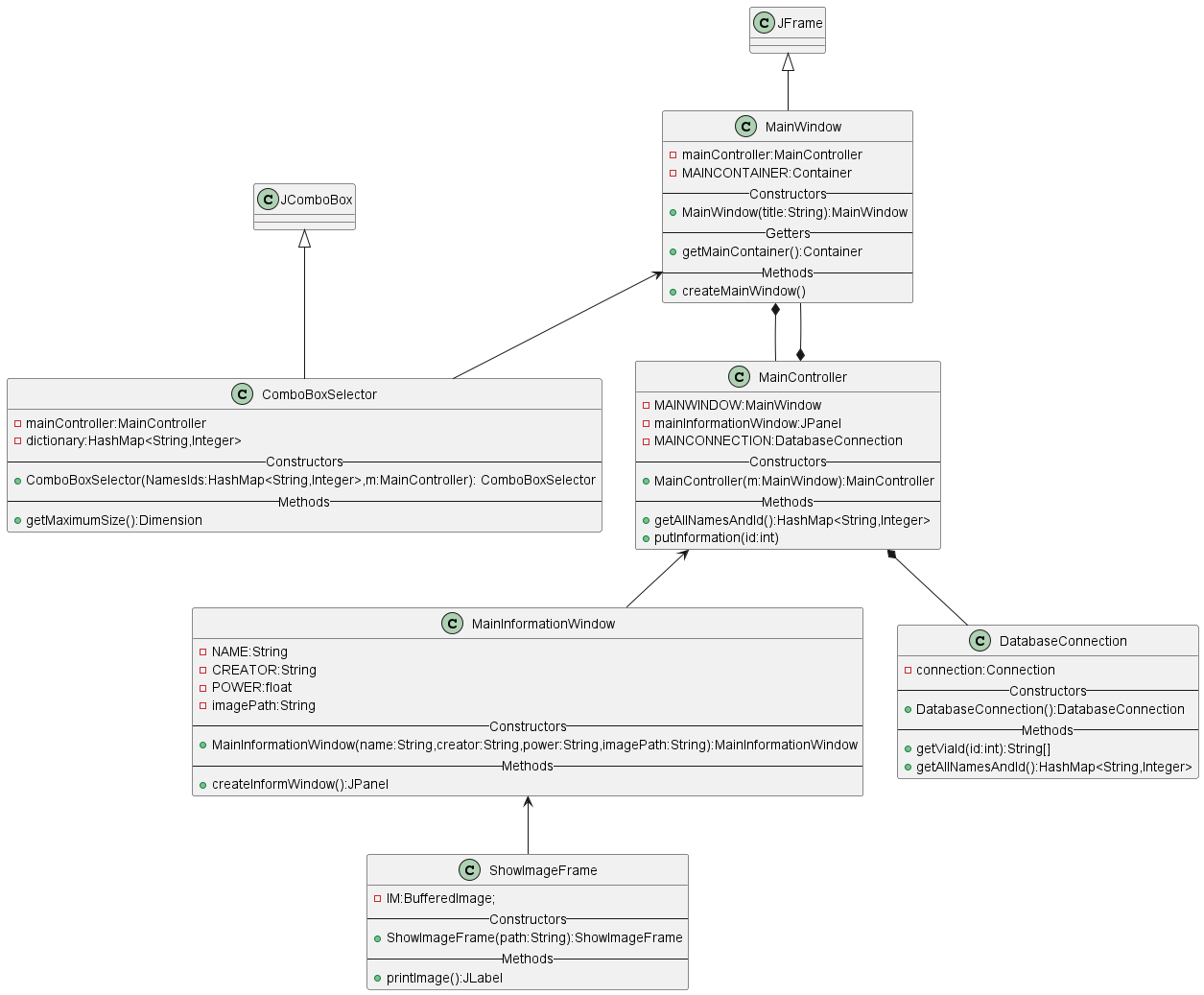


Рисунок 6 – Диаграмма классов приложения.

Для данного приложения была построена диаграмма последовательности, отображающая обработку события выбора машины.

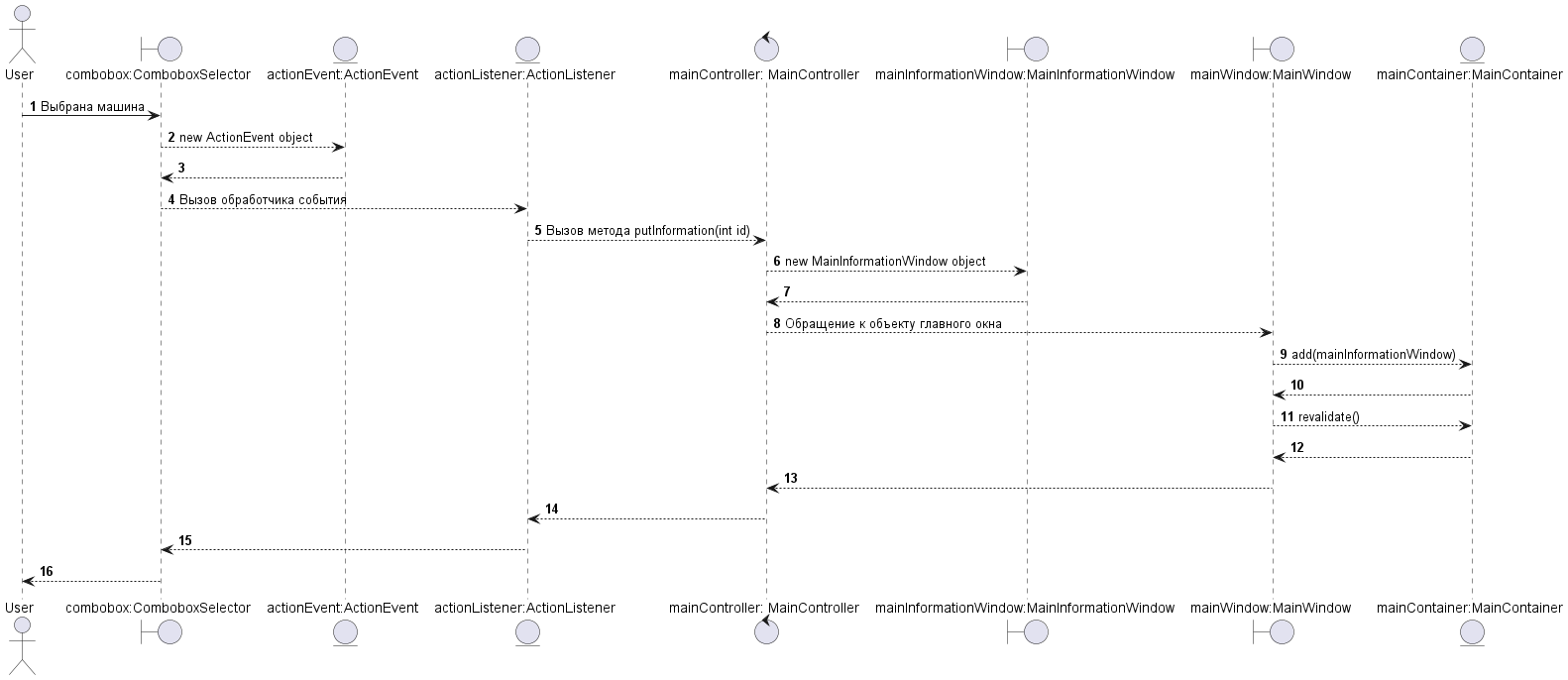


Рисунок 7 – Диаграмма последовательности приложения.

Внешний вид программы имеет вид, представленный на рисунке 8. Для выбора машин был реализован выпадающий список через компонент combobox.

Для хранения информации о машинах была создана таблица БД с помощью sqlite3. C этой целью был написан отдельный класс с методами для подключения к БД, получения информации обо всех или об одном объекте.

При выборе очередной машины из выпадающего списка отображается фотография данной машины, полное название ее модели, ее производитель и мощность.

Полный код программы представлен в приложении 1.



Рисунок 8 – Внешний вид программы.

**ПРИЛОЖЕНИЕ**

**Полный код классов:**

import java.awt.\*;  
  
  
public class Main {  
 public static void main(String[] args) {  
 EventQueue.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 MainWindow m = new MainWindow("Автосалон");  
 m.createMainWindow();  
 }  
 });  
 }  
}

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.HashMap;  
import java.util.Vector;  
  
public class ComboBoxSelector extends JComboBox {  
 MainController mainController;  
 HashMap<String,Integer>dictionary;  
 public ComboBoxSelector(HashMap<String,Integer>NamesIds,MainController m){  
  
  
 super(NamesIds.keySet().toArray(new String[0]));  
  
 this.dictionary = NamesIds;  
 this.mainController = m;  
  
 this.setFont(this.getFont().deriveFont((float)17));  
 ((JLabel)this.getRenderer()).setHorizontalAlignment(SwingConstants.*CENTER*);  
  
 this.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 mainController.putInformation(dictionary.get(getSelectedItem()));  
  
 }  
 });  
  
 }  
 @Override  
 public Dimension getMaximumSize() {  
 Dimension max = super.getMaximumSize();  
 //max.height = getPreferredSize().height;  
 max.height=80;  
 return max;  
 }  
}

import java.sql.\*;  
import java.util.HashMap;  
  
public class DatabaseConnection {  
 private Connection connection;  
 public DatabaseConnection(){  
 this.connection = null;  
  
 try {  
 Class.*forName*("org.sqlite.JDBC");  
 this.connection = DriverManager.*getConnection*("jdbc:sqlite:./Sqlite/test.db");  
 } catch ( Exception e ) {  
 System.*err*.println( e.getClass().getName() + ": " + e.getMessage() );  
 System.*exit*(0);  
 }  
 System.*out*.println("Opened database successfully");  
 }  
 public String[] getViaId(int id){  
 String sql = "SELECT \* FROM Cars WHERE ID=" + String.*valueOf*(id);  
 try {  
  
 Statement stmt = this.connection.createStatement();  
 ResultSet rs = stmt.executeQuery(sql);  
 return new String[]{rs.getString("Name"),rs.getString("Creator"),  
 rs.getString("Power"),rs.getString("Image")};  
  
 } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
 return new String[0];  
 }  
 public HashMap<String,Integer>getAllNamesAndId(){  
 HashMap<String,Integer> dict = new HashMap<>();  
 String sql = "SELECT \* FROM Cars ORDER BY ID ASC";  
 try {  
  
 Statement stmt = this.connection.createStatement();  
 ResultSet rs = stmt.executeQuery(sql);  
 while(rs.next()){  
 dict.put(rs.getString("Name"),rs.getInt("ID"));  
 }  
 } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
 return dict;  
 }  
}

import javax.swing.\*;  
import java.util.Arrays;  
import java.util.HashMap;  
  
public class MainController {  
 private final MainWindow mainWindow;  
 private JPanel mainInformationWindow;  
 private final DatabaseConnection mainConnection;  
 public MainController(MainWindow m){  
 this.mainWindow = m;  
 this.mainConnection = new DatabaseConnection();  
 }  
 public HashMap<String,Integer>getAllNamesAndId(){  
 return mainConnection.getAllNamesAndId();  
 }  
 public void putInformation(int id){  
 System.*out*.println(Arrays.*toString*(mainConnection.getViaId(id)));  
 if(mainInformationWindow!=null) mainWindow.getMainContainer().remove(mainInformationWindow);  
 this.mainInformationWindow = new MainInformationWindow(mainConnection.getViaId(id)[0],  
 mainConnection.getViaId(id)[1],mainConnection.getViaId(id)[2],mainConnection.getViaId(id)[3]).createInformWindow();  
 mainWindow.getMainContainer().add(mainInformationWindow);  
 mainWindow.getMainContainer().revalidate();  
 }  
}

import javax.swing.\*;  
import java.awt.\*;  
  
public class MainInformationWindow {  
 private final String name;  
 private final String creator;  
 private final float Power;  
 private final String imagePath;  
 public MainInformationWindow(String name, String creator, String power, String imagePath){  
 this.name=name;  
 this.creator=creator;  
 this.Power= Float.*parseFloat*(power);  
 this.imagePath=imagePath;  
 }  
 public JPanel createInformWindow(){  
 JPanel j =new JPanel(new FlowLayout());  
 Container c = new Container();  
  
  
 c.setLayout(new GridLayout(3,1,5,5));  
  
  
 JLabel l1 = new JLabel("Модель:"+this.name);  
 l1.setFont(l1.getFont().deriveFont((float)20));  
 c.add(l1);  
  
 l1 = new JLabel("Производитель:"+this.creator);  
 l1.setFont(l1.getFont().deriveFont((float)17));  
 c.add(l1);  
  
 l1 = new JLabel("Мощность:"+String.*valueOf*(this.Power)+" л.c.");  
 l1.setFont(l1.getFont().deriveFont((float)17));  
 c.add(l1);  
  
 j.add(new ShowImageFrame(imagePath).printImage());  
 j.add(c);  
  
  
 return j;  
 }  
}

import javax.imageio.ImageIO;  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ComponentAdapter;  
import java.awt.event.ComponentEvent;  
import java.awt.image.BufferedImage;  
import java.io.File;  
import java.io.IOException;  
  
public class MainWindow extends JFrame {  
 private final MainController mainController;  
 private Container mainContainer;  
  
  
 public Container getMainContainer() {  
 return mainContainer;  
 }  
  
 public MainWindow(String title){  
 super(title);  
 this.mainController = new MainController(this);  
  
 }  
 public void createMainWindow(){  
  
 setSize(1000,600);  
 setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);  
  
  
 this.mainContainer = getContentPane();  
 BoxLayout boxLayout = new BoxLayout(mainContainer,BoxLayout.*Y\_AXIS*);  
 mainContainer.setLayout(boxLayout);  
  
 ComboBoxSelector c = new ComboBoxSelector(this.mainController.getAllNamesAndId(),this.mainController);  
  
 mainContainer.add(c);  
 c.setSelectedIndex(1);  
  
 setVisible(true);  
 }  
}

import javax.imageio.ImageIO;  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.image.BufferedImage;  
import java.io.File;  
import java.io.IOException;  
  
public class ShowImageFrame {  
 private final BufferedImage im;  
 public ShowImageFrame(String path){  
 try {  
 im = ImageIO.*read*( new File(path));  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 }  
 public JLabel printImage(){  
 ImageIcon imageIcon = new ImageIcon(this.im);  
 Image image = imageIcon.getImage(); // transform it  
 Image newimg = image.getScaledInstance(500, 500, java.awt.Image.*SCALE\_SMOOTH*); // scale it the smooth way  
 imageIcon = new ImageIcon(newimg); // transform it back  
 JLabel ma = new JLabel(imageIcon);  
 ma.setPreferredSize(new Dimension(500,500));  
 return ma;  
 }  
}