Министерство образования и науки Украины

Одесский Политехнический Университет

Кафедра “Информационные технологии”

Отчёт по лабораторной работе №8.1

На тему: «Создание графического интерфейса пользователя»

Выполнил cтудент группы:

АД-201

Стыцковский Н.Ю

Проверили:

Рудниченко Н.Д.

Оглавление

[Цель работы 2](#_Toc86140774)

[Теоретические сведения 2](#_Toc86140775)

[Листинг 4](#_Toc86140776)

[Main 4](#_Toc86140777)

[Controller 5](#_Toc86140778)

[**Висновок:** 6](#_Toc86140779)

Цель работы: Научиться создавать автономные приложения для работы с окнами и элементами меню.

## Теоретические сведения

В Java есть три библиотеки визуальных компонентов для создания графического интерфейса пользователя.

Самая ранняя из библиотек для создания графического интерфейса пользователя называется AWT (Abstract Window Toolkit). Библиотечные методы AWT создают и используют графические компоненты операционной среды. С одной стороны, это хорошо, так как программа на Java похожа на остальные программы в рамках одной ОС. Но при запуске ее на другой платформе могут возникнуть различия в размерах компонентов и шрифтов, которые будут портить внешний вид программы.

Вслед за AWT была разработана библиотека Swing. Она создана на базе AWT и заменяет большинство ее компонентов своими, спроектированными более тщательно и удобно.

Каждая библиотека предоставляет набор классов для работы с кнопками, списками, окнами, меню и т.д., но эти классы спроектированы по-разному: они имеют различный набор методов с разными параметрами.

Библиотека AWT.

Использование класса Frame для реализации окна приложения

Класс Frame представляет собой подкласс Window, инкапсулирующий окно приложения.

Объект Frame может содержать строку меню и отображать заголовок.

Методы класса Frame:

|  |  |
| --- | --- |
| Метод | Описание |
| void pack(); | Изменение размер компонентов в окне так, чтобы их размер был максимально приближен к желаемому. |
| void setTitle(String s); | Установка заголовка окна. |
| void setMenuBar(MenuBar menubar); | Установка меню окна. |
| void remove(MenuComponent component); | Удаление компонента из меню окна. |
| void setResizable(boolean); | Разрешение изменения размеров окна. |
| void setCursor(int d); | Установка типа курсора мыши для окна. |

Использование класса MenuBar для реализации меню

Класс MenuBar реализует строки меню. Он является подклассом класса MenuComponent, содержащего общий набор методов для всех классов, связанных с меню. Объект MenuBar можно присоединить к объекту Frame, используя метод setMenuBar() класса Frame.

Объект MenuBar содержит один или несколько объектов Menu, реализующих ниспадающие меню. Класс Menu включает методы, обеспечивающие добавление к ниспадающему меню объектов MenuItem и разделителей. Кроме того, он предлагает методы для доступа к объектам MenuItem, которые уже находятся в объекте Menu. Поскольку класс Menu является подклассом MenuItem, объект Menu может содержать другой объект Menu, так что возможно создание каскадных меню с несколькими уровнями.

Использование классов Label и Button для реализации меток и кнопок в программе

Класс Label отображает метку – строку текста, которую нельзя редактировать. Методы класса Label:

|  |  |
| --- | --- |
| Метод | Описание |
| public void setAlignment(int alignment); | Установка выравнивания текстового поля |
| public void setText(String label); | Запись текста в поле |

Класс Button позволяет выводить кнопки в окнах Java-приложений и апплетов. Методы класса Button:

|  |  |
| --- | --- |
| Метод | Описание |
| public String getLabel(); | Получение надписи на кнопке |
| public void setLabel(String label); | Создание надписи на кнопке |

Любой компонент, требующий показа на экране, должен быть добавлен в класс-контейнер. Контейнеры служат хранилищем для визуальных компонентов интерфейса и других контейнеров. Простейший пример контейнера − класс Frame. Еще один пример контейнера – класс Panel.

Чтобы добавить компонент в контейнер, применяется перегруженный метод add:

* add(someThing); // Вставить элемент после других элементов в контейнере
* add(-1, someThing); // Вставить элемент в окно контейнера у его верхней границы
* add("Расположение", someThing); // Вставить элемент в определенной части контейнера: North, South, East, West или Center.

## Листинг

## App

## package com.mycompany.lab.stytskovskyi;

## import javafx.application.Application;

## import javafx.fxml.FXMLLoader;

## import javafx.scene.Parent;

## import javafx.scene.Scene;

## import javafx.stage.Stage;

## import java.io.IOException;

## /\*\*

## \* JavaFX App

## \*/

## public class App extends Application {

## private static Scene scene;

## @Override

## public void start(Stage stage) throws IOException

## {

## scene = new Scene(loadFXML("primary"), 640, 480);

## stage.setScene(scene);

## stage.show();

## }

## static void setRoot(String fxml) throws IOException {

## scene.setRoot(loadFXML(fxml));

## }

## private static Parent loadFXML(String fxml) throws IOException {

## FXMLLoader fxmlLoader = new FXMLLoader(App.class.getResource(fxml + ".fxml"));

## return fxmlLoader.load();

## }

## public static void main(String[] args)

## {

## launch();

## }

## }

## Controller

package com.mycompany.lab.stytskovskyi;

import javafx.scene.input.MouseEvent;

import javafx.scene.control.Label;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.control.ChoiceBox;

import javafx.scene.control.TextField;

import javafx.scene.control.RadioButton;

public class PrimaryController

{

private String noun;

private String verb;

private String adverb;

private String inputString;

@FXML

private ChoiceBox<String> adverbSelect;

@FXML

private ChoiceBox<String> pronounSelect;

@FXML

private RadioButton radioAdverb;

@FXML

private RadioButton radioNoun;

@FXML

private RadioButton radioParticle;

@FXML

private TextField textInput;

@FXML

private Label textOutput;

@FXML

public void execute (ActionEvent event)

{

textOutput.setText("");

noun = "";

verb = "";

adverb = "";

inputString = "";

inputString = textInput.getText();

for (int i = 0, j = 1; i < inputString.length(); i++) {

if (inputString.charAt(i) == ' ') {

j++;

continue;

}

if (j == 1)

noun += inputString.charAt(i);

if (j == 2)

verb += inputString.charAt(i);

if (j == 3)

adverb += inputString.charAt(i);

}

if (radioNoun.isSelected()==true)

{

noun = pronounSelect.getValue();

if ("Она".equals(pronounSelect.getValue()))

{

verb += 'a';

}

else if ("Они".equals(pronounSelect.getValue()))

{

verb += 'и';

}

}

if (radioParticle.isSelected() == true)

{

verb += " бы";

}

if (radioAdverb.isSelected() == true)

{

adverb = adverbSelect.getValue();

}

textOutput.setText(noun + ' ' + verb + ' ' + adverb);

}

@FXML

void enableAdverbSelect(ActionEvent event)

{

adverbSelect.disableProperty().set((adverbSelect.disableProperty().get() != true));

}

@FXML

void enablePronounSelect(ActionEvent event)

{

pronounSelect.disableProperty().set((pronounSelect.disableProperty().get() != true));

}

@FXML

public void pronounFill(MouseEvent event)

{

if (pronounSelect.getItems().size() < 2)

pronounSelect.getItems().addAll("Он", "Она", "Они");

}

@FXML

public void adverbFill(MouseEvent event)

{

if (adverbSelect.getItems().size() < 2)

adverbSelect.getItems().addAll("здесь", "везде", "нигде", "зачем?", "почему?");

}

@FXML

public void clearInput(ActionEvent event)

{

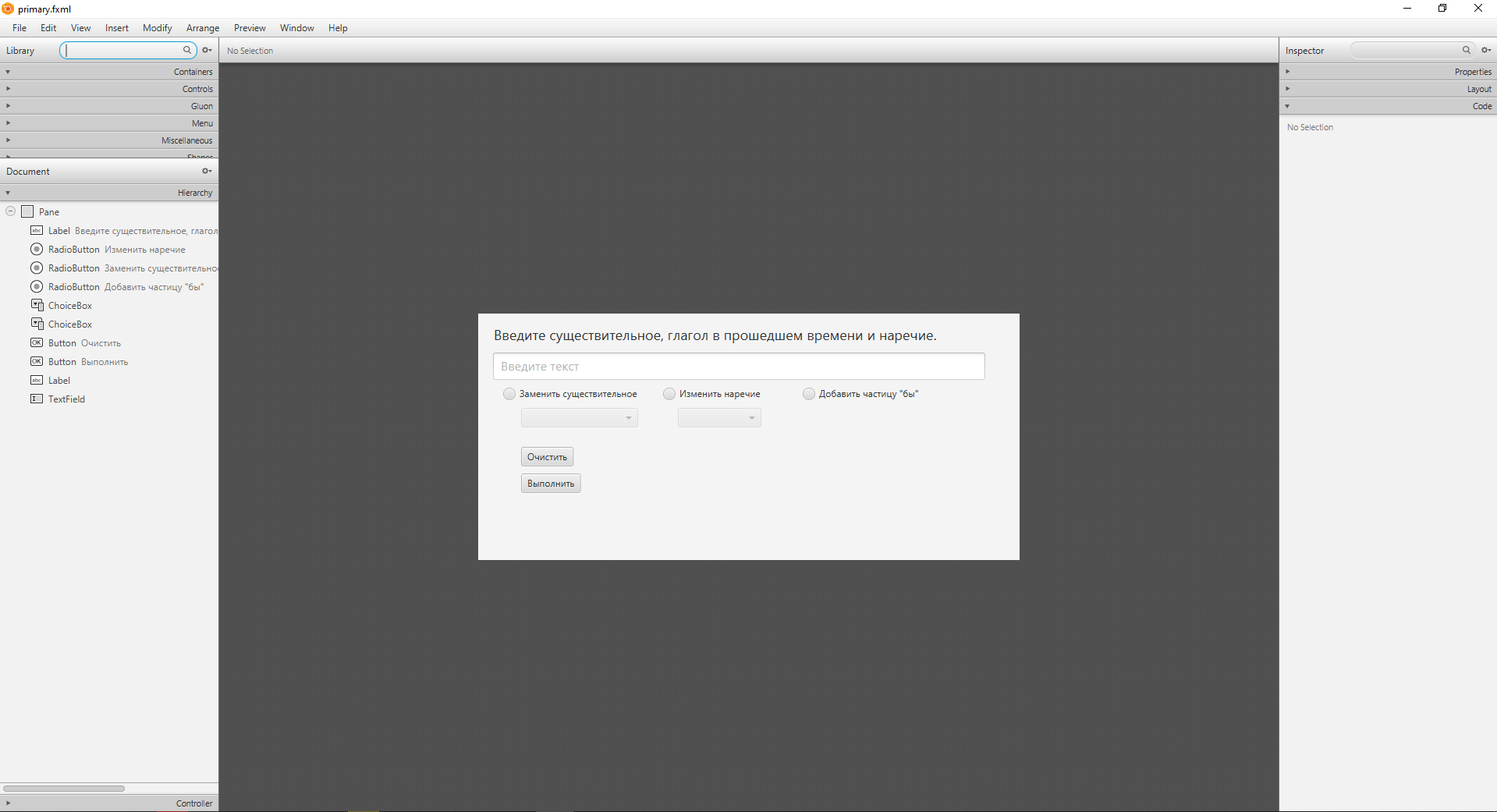
textInput.clear();

textOutput.setText("");

}

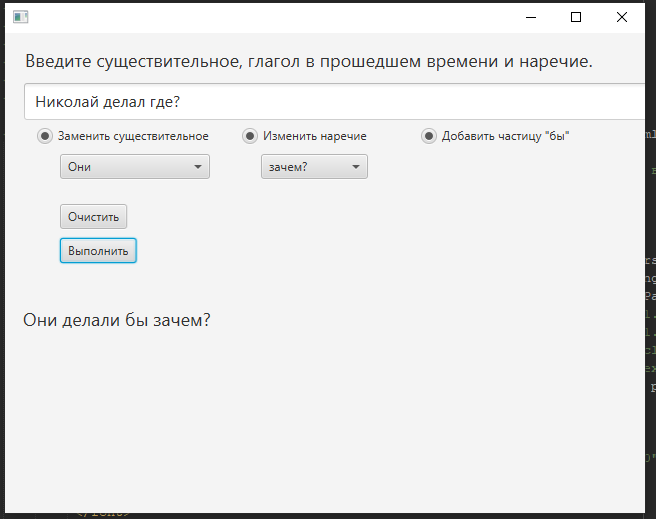
}

Scene Builder



Скриншот 1. Вид программы.

Скриншот



## Вывод

В ходе выполненной работы мы научились пользоваться JavaFX, смогли создать графический интерфейс с разными операциями.