

Звіт

Автор: Васильченко С., 1.КІТ1186

Дата: 10.02.2020

Лабораторна робота №16

РОЗРОБКА ГРАФІЧНОГО ІНТЕРФЕЙСУ КОРИСТУВАЧА

Мета:

Придбання навичок використання засобів клієнтських технологій (*Client Technologies*) платформи *Java SE*.

Вимоги:

Розробити графічний інтерфейс користувача для програми рішення попередньої лабораторної роботи з використанням засобів *JavaFX*.

ПРИКЛАДНА ЗАДАЧА:

Кадрове агентство. Сортуння за назвою фірми, за назвою запропонованої спеціальності, за вказаною освітою.

ОПИС ПРОГРАМИ

2.1 Опис змінних:

```
LinkedList<Recruitment> stringLinked = new LinkedList<>(); // об'єкт  
параметризованого контейнера
```

```
Recruitment rec1 = new Recruitment(); // об'єкт класа кадрового агенства
```

```
Scanner scan = new Scanner(System.in); // змінна для активування  
зчитування з консолі
```

2.2 Ієрархія та структура класів.

final class Lab15 – головний клас. Містить метод main(точку входу у програму) та методи по роботі з програмою для реалізації індивідуального завдання.

Class Container – клас реалізований на LinkedList

class Recruitment - клас прикладної задачі кадрового агенства

class Util - клас зберігаючий утиліти для обробки контейнера

class RunProgramm – клас запускаючий на виконання першу сцену, яка далі буде запускати додаткові сцени

ТЕКСТ ПРОГРАМИ

File Main.java:

```
package ua.khpi.oop.vasilchenko16.App;

import ua.khpi.oop.vasilchenko16.Container.Container;

public class Main {
    public static Container container = new Container();
}
```

RunProgramm.java:

```
package ua.khpi.oop.vasilchenko16.App;

import javafx.application.Application;
import javafx.stage.Stage;
import ua.khpi.oop.vasilchenko16.Util.Util;

public class RunProgram extends Application {

    public static void main(String[] args) {
        launch(args);
    }

    @Override
    public void start(Stage primaryStage) {
        Util.createNewScene("welcome.fxml", "Recruitment");
    }
}
```

Recruitment.java :

```
package ua.khpi.oop.vasilchenko09.First;

import java.util.Scanner;

public class Recruitment {
    private String firm;
    private String specialty;
    private String workingConditions;
    private int payment;

    private String needsSpeciality;
    private int experience;
    private String education;

    private boolean confirm ;

    public void setFirm(final String firm) {
        this.firm = firm;
    }

    public void setSpecialty(final String specialty) {
        this.specialty = specialty;
    }
}
```

```

public void setWorkingConditions(final String workingConditions) {
    this.workingConditions = workingConditions;
}

public void setPayment(final int payment) {
    this.payment = payment;
}

public void setConfirm(final boolean confirm) {
    this.confirm = confirm;
}

public int getPayment() {
    return payment;
}

public String getWorkingConditions() {
    return workingConditions;
}

public String getSpecialty() {
    return specialty;
}

public String getFirm() {
    return firm;
}

public Recruitment() {
    firm = null;
    specialty = null;
    workingConditions = null;
    payment = 0;
    needsSpeciality = null;
    experience = 0;
    education = null;
    confirm = false;
}

public String show() {
    String show;
    show = "Фирма: " + firm + "\n" +
        "Специальность: " + specialty + "\n" +
        "Условия работы: " + workingConditions + "\n" +
        "Оплата: " + payment + "\n";
    if (confirm) {
        show += "Необходимая специальность: " + needsSpeciality + "\n";
        show += "Опыт: " + experience + "\n";
        show += "Образование: " + education + "\n";
    }
    return show;
}

public Recruitment(final Recruitment obj) {
    firm = obj.firm;
    specialty = obj.specialty;
    workingConditions = obj.workingConditions;
    payment = obj.payment;
    needsSpeciality = obj.needsSpeciality;
    experience = obj.experience;
    education = obj.education;
    confirm = obj.confirm;
}

public void generateVacancy() {
    Scanner scan = new Scanner(System.in);
    Scanner scan2 = new Scanner(System.in);

```

```

        int choose;
        System.out.print("\nВведите фирму: ");
        firm = scan.nextLine();
        System.out.print("\nВведите специальность: ");
        specialty = scan.nextLine();
        System.out.print("\nВведите условия работы: ");
        workingConditions = scan.nextLine();
        System.out.print("\nВведите оплату: ");
        payment = scan.nextInt();
        System.out.println("Желаете добавить дополнительные условия работы? 1 - Да. 0 - Нет:");
    });

    choose = scan.nextInt();
    while (true) {
        if (choose == 1) {
            System.out.print("\nВведите необходимую специальность: ");
            needsSpeciality = scan2.nextLine();
            System.out.print("\nНеобходимое образование: ");
            education = scan2.nextLine();
            System.out.print("\nнеобходимый опыт работы: ");
            experience = scan2.nextInt();
            confirm = true;
            break;
        } else if (choose == 0) {
            needsSpeciality = null;
            experience = 0;
            education = null;
            break;
        } else {
            System.out.println("Ошибка! Повторите ввод: ");
        }
    }
}

public void setExperience(final int experience) {
    this.experience = experience;
}

public int getExperience() {
    return experience;
}

public void setNeedsSpeciality(final String needsSpeciality) {
    this.needsSpeciality = needsSpeciality;
}

public String getNeedsSpeciality() {
    return needsSpeciality;
}

public void setEducation(final String education) {
    this.education = education;
}

public String getEducation() {
    return education;
}

public boolean getConfirms() {
    return confirm;
}

@Override
public String toString() {
    return show();
}
}

```

Util.java:

```
package ua.khpi.oop.vasilchenko16.Util;

import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.stage.Stage;
import ua.khpi.oop.vasilchenko16.App.Main;
import ua.khpi.oop.vasilchenko16.Container.Container;
import ua.khpi.oop.vasilchenko16.Controllers.ControllerWelcome;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

public class Util {

    public static void save() throws IOException {
        FileOutputStream outputStream = new
FileOutputStream("src/ua/khpi/oop/vasilchenko16/Save/data.bin");
        ObjectOutputStream objectOutputStream = new ObjectOutputStream(outputStream);
        objectOutputStream.writeObject(Main.container);
        objectOutputStream.close();
    }

    public static void read() throws IOException, ClassNotFoundException {
        FileInputStream fileInputStream = new
FileInputStream("src/ua/khpi/oop/vasilchenko16/Save/data.bin");
        ObjectInputStream objectInputStream = new ObjectInputStream(fileInputStream);
        Main.container = (Container) objectInputStream.readObject();
    }

    public static void createNewScene(String fxml, String name){
        FXMLLoader loader = new FXMLLoader();

loader.setLocation(ControllerWelcome.class.getResource("/ua/khpi/oop/vasilchenko16/View/" +
fxml));
        try {
            loader.load();
        } catch (IOException e) {
            e.printStackTrace();
        }
        Parent root = loader.getRoot();
        Stage stage = new Stage();
        stage.getIcons().add(new
Image("file:/ua/khpi/oop/vasilchenko16/Assist/customer_person_people_man_you_1625.ico"));
        stage.setTitle(name);
        stage.setScene(new Scene(root));
        stage.show();
    }
}
```

Сцени релізовані за допомогою програмного забезпечення SceneBuilder та fxml файлів.

ВАРІАНТИ ВИКОРИСТАННЯ



Рис. 16.1 – Результат роботи програми

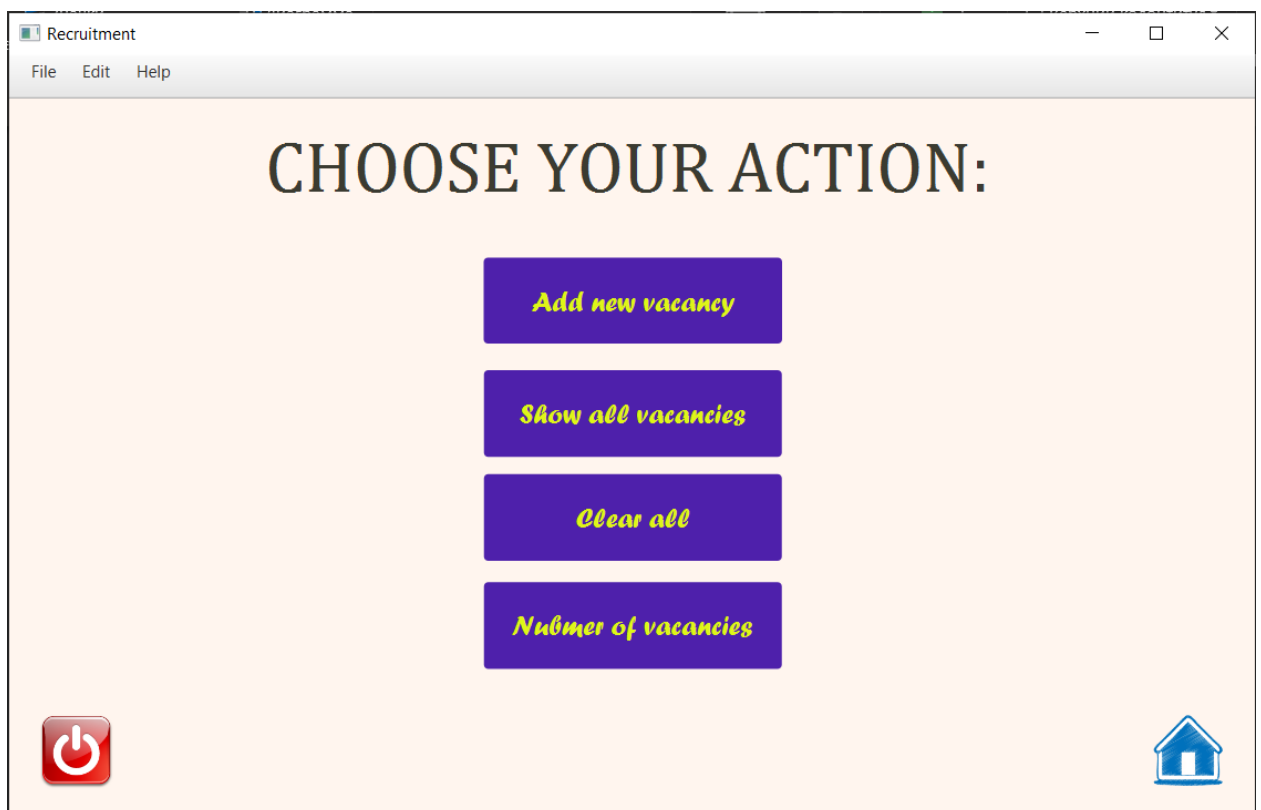


Рис. 16.2 – Результат роботи програми

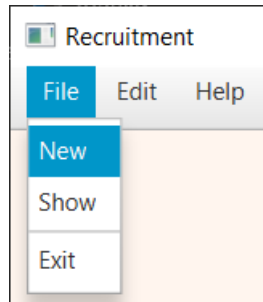


Рис. 16.3 – Результат роботи програми

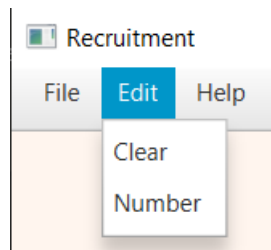


Рис. 16.4 – Результат роботи програми

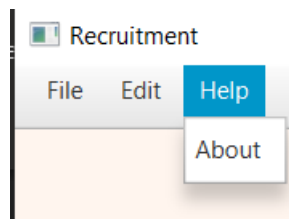


Рис. 16.5 – Результат роботи програми



Рис. 16.6 – Результат роботи програми

Recruitment

File Edit

ENTER THE DATA:


Specialty

Working conditions

Payment

Additional Conditions:

☐ Yes ☒ No



Save

Рис. 16.7 – Результат работы программы

Additional Conditions:

☒ Yes ☐ No

Necessary specialty

Education

Experience

Save

Рис. 16.8 – Результат работы программы

JD:

Search

Рис. 16.12 – Результат роботи програми

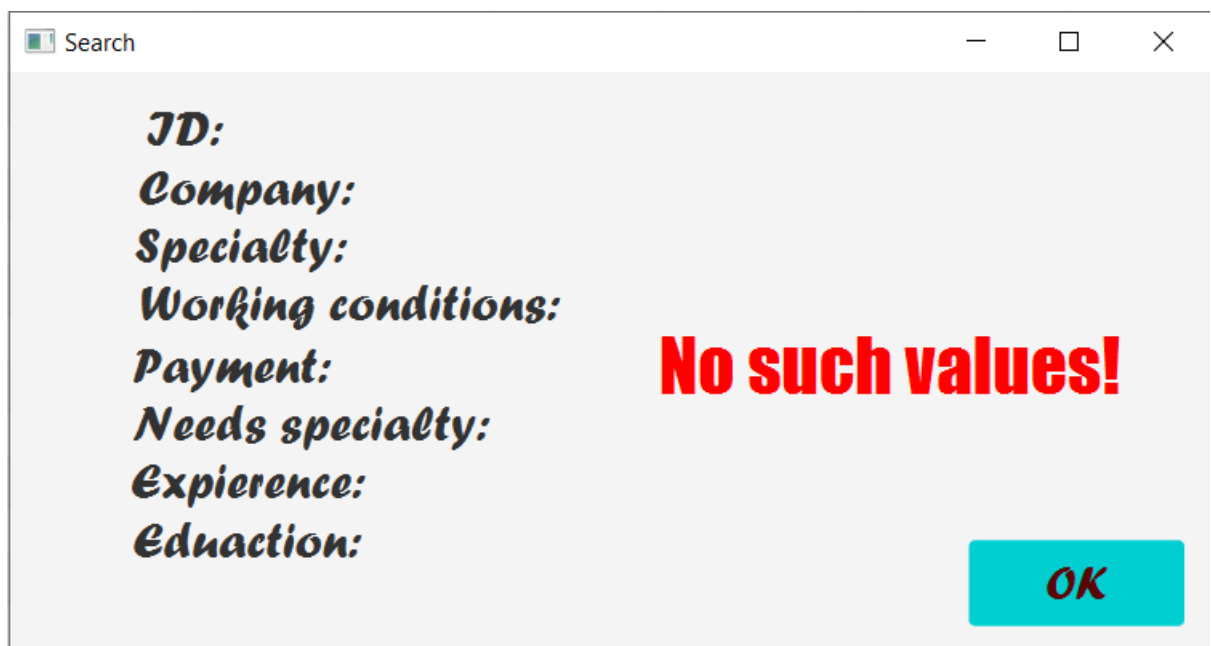


Рис. 16.13 – Результат роботи програми

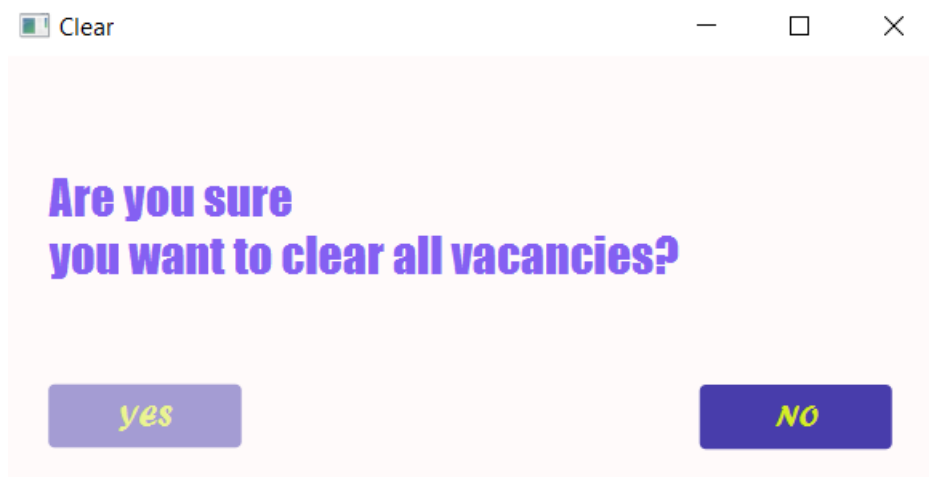


Рис. 16.16 – Результат роботи програми

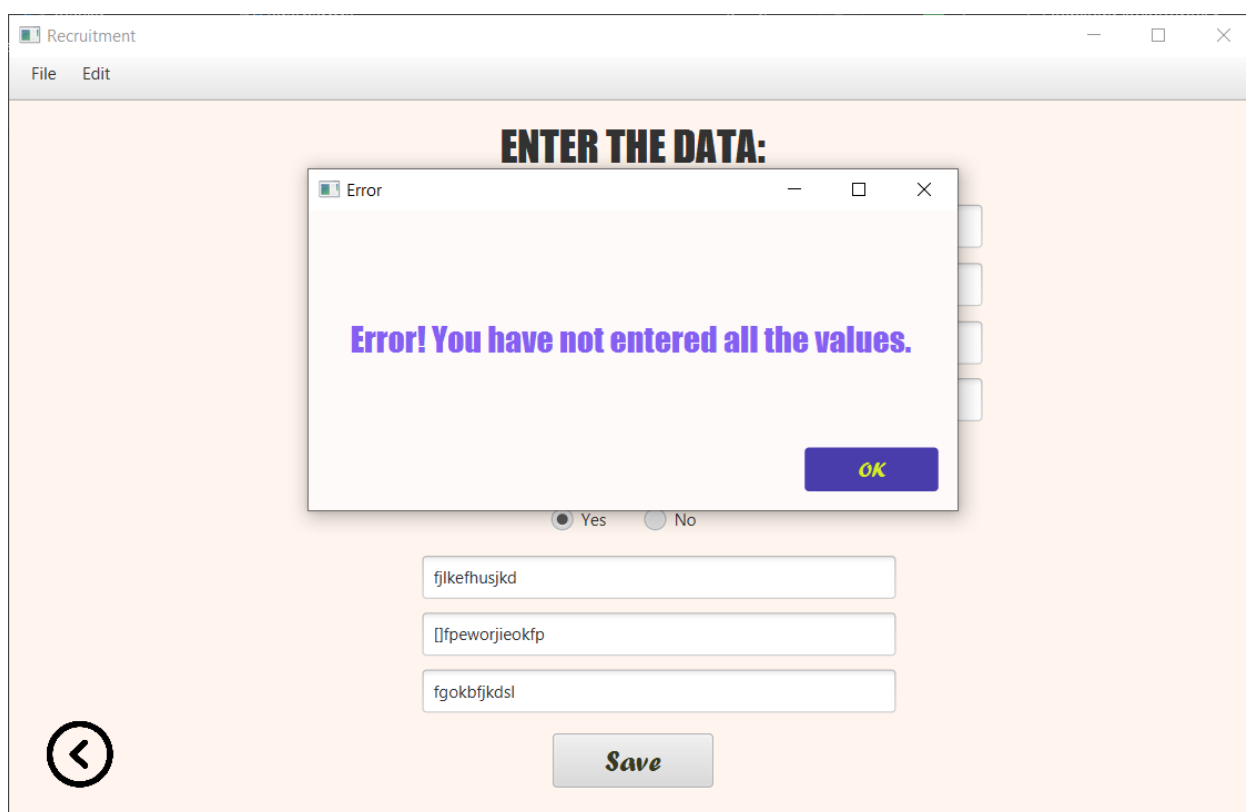


Рис. 16.17 – Результат роботи програми

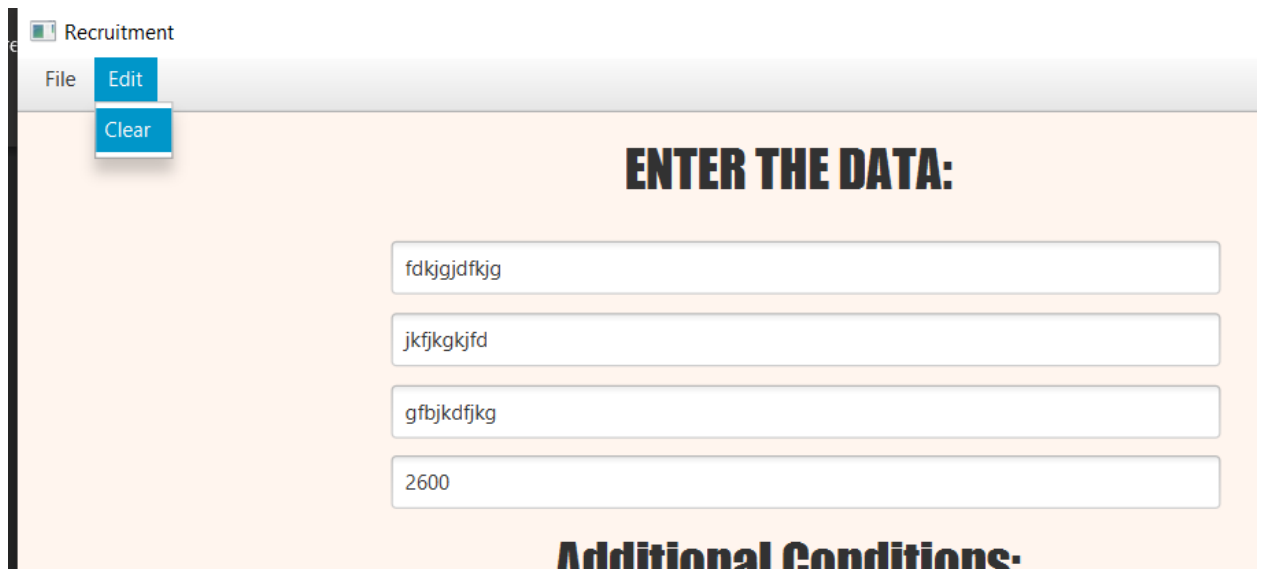


Рис. 16.18 – Результат роботи програми

Програму можна використовувати задля створення бази даних. Завдяки параметризації зв'язного списку, базу даних можна використати для будь-яких типів даних. Переважно у нашому варіанті - кадрове агенство, в якому представляються різноманітні вакансії. Також для вибору доступно багато інших можливостей. Реалізовано меню для поліпшення користування програмою. Реалізовано графічний інтерфейс для поліпшення користування програмою та підвищення продуктивності роботи з базою даних.

ВИСНОВКИ

При виконанні лабораторної роботи набуто практичних навичок щодо використання параметризованих класів. Розроблено графічний інтерфейс користувача для програми рішення попередньої лабораторної роботи з використанням засобів JavaFX. Також навчився обробляти параметризовані контейнери. Завдання виконане! Програма працює успішно!