Лабораторна робота № 12

Знайомство з бібліотекою візуальних компонентів Swing.

Мета роботи: Ознайомлення із засобами Java, призначеними для створення графічного інтерфейсу користувача та методикою створення додатків, що реагують на події.

Короткі теоретичні відомості.

Бібліотека візуальних компонентів Swing

Графічний інтерфейс (GUI), завдяки своїй зручності, став головним способом реалізації діалогу між користувачем та програмою. Дії, які може виконувати програма, зібрані у вигляді набору пунктів графічного меню вікна програми, для часто використовуваних команд виводяться візуальні кнопки на панелях інструментів чи візуальних закладках, параметри виконання команд задаються за допомогою графічних елементів керування у вікнах діалогу і т.п.

В перших версіях Java для створення компонентів GUI було використано спеціальний пакет Abstract Window Toolkit (AWT), що не завоював популярності серед програмістів. Тоді AWT насправді не виконували ніякої роботи і були дуже простими - це були просто «Java-оболонки» для елементів управління тієї операційної системи, на якій працював користувач. Всі запити до цих компонентів непомітно перенаправлялись до операційній системі, яка і виконувала всю роботу. Щоб зробити класи AWT незалежними від конкретної платформи, кожному з них був зіставлений своєрідний помічник (peer), який і працював з цією платформою. Для того щоб вбудувати в AWT підтримку нової платформи, потрібно було просто переписати код цих помічників, а інтерфейс основних класів залишався незмінним. Такий підхід виявився не досить вдалим як з точки зору швидкодії програм, так із точки зору гнучкості та можливостей дизайну інтерфейсу. В пакети AWT наступних версій Java було внесено принципові зміни – пакет доповнили так званими «легкими» («легковаговими» - lightweight) компонентами функціонування яких зав'язане не на конкретну операційну систему а на віртуальну машину Java. Як результат – значний виграш у швидкодії програм та майже необмежені можливості графічного дизайну. Раніше розроблені «важкі» компоненти AWT залишилися в пакеті для забезпечення взаємодії «легких» компонентів з операційною системою. Згодом, на основі вдосконалених компонентів AWT, було розроблено ще одну бібліотеку «легких» графічних компонентів – Swing. Компоненти Swing набули популярності і знайшли широке використання у розробці візуальних інтерфейсів сучасних Java-додатків.

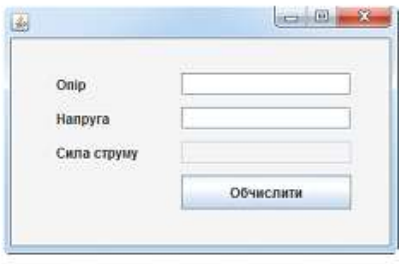
Завдання. Розробити простий графічний інтерфейс для розв'язання прикладної обчислювальної задачі та реалізувати Java-програму для його обслуговування. Інтерфейс повинен включати вікно програми (фрейм), текстові поля для вводу вхідних даних, поле для виводу результату та кнопку для запуску обчислення (див. приклад виконання).

****Додаткове завдання.*** Модифікувати створену програму додавши можливість розв'язування обернених задач. Тобто, дозволити ввід даних у поле для результату обчислень і запрограмувати наступну поведінку вікна: як тільки введених у полях даних достатньо для розв'язання однієї із можливих задач, поле без даних робиться недоступним, а кнопка повинна обчислювати відповідну величину. Наприклад, якщо у задачі про закон Ома ввести опір R та силу струму I, то програма буде обчислювати напругу U, а якщо задати силу струму I та напругу U, то програма обчислюватиме опір і т.д., при цьому після вводу даних у будь-яких два поля третє повинно автоматично блокуватися для виводу результату.

Варіанти завдань

1. Знайти максимальну висота польоту тіла випущеного з початковою швидкістю v_0 під кутом $0^0 < \alpha < 90^0$ до горизонту: $h_{\max} = \frac{(v_0 \sin\alpha)^2}{2g}$.
2. Визначити тиск стовпа висотою h рідини, що має густину ρ: $p = \rho g h$.
3. Знайти потенціальну енергію V $м^3$ води, піднятих греблею на висоту h м.: $W_n = \rho V g h$. Густину води вважати рівною 1000 кг/ $м^3$.
4. За законом Архімеда розрахувати умову плавання тіла масою m та об'ємом V, зануреного у рідину з густиною ρ.
5. Визначити число ядер радіоактивної речовини, що розпалися за період часу t, якщо відомо сталу радіоактивного розпаду λ та початкове число ядер N_0.
6. Обчислити величину роботи з переміщення тіла на відстань S, виконаною силою F, прикладеною у напрямку переміщення.
7. Встановити кількість теплоти потрібну для нагрівання m кг заданої речовини на t градусів.
8. Встановити величину сили F, яку потрібно прикласти до пружини заданої жорсткості, щоб стиснути її на вказану величину.
9. Знайти нормальне прискорення тіла, що рухається по колу радіуса R зі сталою лінійною швидкістю v.
10. Визначити швидкість поширення звуку в рідині з модулем усестороннього тиску K та густиною ρ.
11. Обчислити ємність плоского конденсатора з площею пластин S та відстанню d між ними.
12. Визначити кількість теплоти, що утворюється від згоряння m кг заданої речовини з питомою теплотою згоряння q Дж/кг.
13. Обчислити потужність механізму, що може виконати роботу A за час t.
14. Знайти кінетичну енергію тіла масою m, що рухається зі швидкістю v: $W_k = \frac{mv^2}{2}$.
15. Визначити відстань, яку пролетить тіло випущене з початковою швидкістю v_0 під кутом $0^0 < \alpha < 90^0$ до горизонту: $h_{\max} = \frac{v_0^2 \sin(2\alpha)}{g}$.
16. Визначити силу струму в провіднику за законом Ома: $I = \frac{U}{R}$.

Приклад виконання лабораторної роботи.
Варіант 16. Закон Ома.



Спосіб 1. Код програми можна записати у звичайному Java проекті у вигляді окремого класу, що наслідує клас JFrame із пакету javax.swing:

```
public class FirstForm extends JFrame {}
```

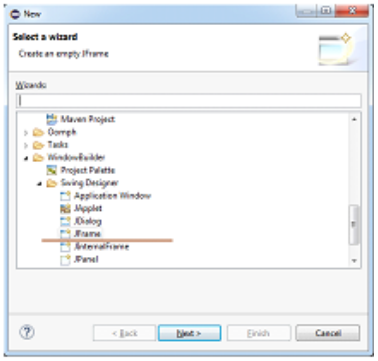
При цьому, очевидно слід імпортувати класи потрібних візуальних компонентів та обробників подій. Код програми для розв'язання задачі може бути приблизно таким (призначення відповідних фрагментів подано у коментарях):

```java
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class FirstForm extends JFrame {

    private static final long serialVersionUID = 1L;
    //Об'єктам типу фрейм прийнято присвоювати номер версії
    //додано підказкою Eclipse
    //оголошення текстових полів для вводу (виводу) даних
    private JTextField textField1;
    private JTextField textField2;
    private JTextField textField3;
    //JPanel використовується як контейнер для розміщення
    //візуальних компонентів на формі
    private JPanel contentPane;

    //конструктор, в якому створюється форма з її компонентами
    public FirstForm() {

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 350, 220);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        contentPane.setLayout(null);
        //обнулення шаблону компоновки елементів форми
        //дозволяє абсолютне позиціювання, тобто можна
        //встановлювати керуючі компоненти задаючи їх координати
        //в будь-яке міце форми

        setContentPane(contentPane);

        //створення об'єктів - елементів керування

        JLabel label1 = new JLabel("Опір");

        textField1 = new JTextField();

        JLabel label2 = new JLabel("Напруга");

        textField2 = new JTextField();

        JLabel label3 = new JLabel("Сила струму");

        textField3 = new JTextField();
```

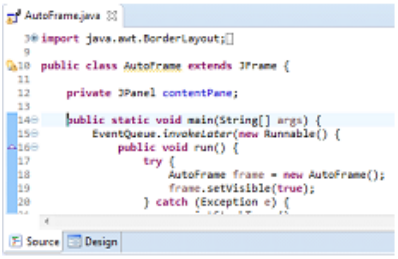
```
        /*розстановка елементів на формі
         *в методі setBounds задають координати
         *верхнього лівого кута елемента відносно форми
         *та його розміри
         *метод add встановлює створений елемент
         *в контейнер форми
         */

        label1.setBounds(40, 30, 80, 20);
        label2.setBounds(40, 60, 80, 20);
        label3.setBounds(40, 90, 80, 20);

        textField1.setBounds(150, 30, 150, 20);
        textField2.setBounds(150, 60, 150, 20);
        textField3.setBounds(150, 90, 150, 20);
            //в поле для результату ввід заборонено
        textField3.setEditable(false);


        contentPane.add(label1);
        contentPane.add(textField1);
        contentPane.add(label2);
        contentPane.add(textField2);
        contentPane.add(label3);
        contentPane.add(textField3);

        //ще один керуючий елемент - кнопка для запуску процесу обчислення
        JButton btnCalculate = new JButton("Обчислити");
        btnCalculate.setBounds(150, 120, 150, 30);
        contentPane.add(btnCalculate);

    //на кнопку btnCalculate підписується "слухач" для відслідковування
        //події - натискання кнопки, в цьому прикладі - це анонімний клас
        //в тілі якого описана реакція форми на натискання кнопки
        btnCalculate.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
                double r = Double.parseDouble(textField1.getText());
                double u = Double.parseDouble(textField2.getText());
                double i = u/r;//обчислення реалізовано безпосередньо
                //в обробнику натискання кнопки
                textField3.setText(Double.toString(i));
            }
        });
        //наприкінці конструктора задається відображення форми на екрані
        setVisible(true);
    }

    public static void main(String[] args) {
        new FirstForm();//запуск форми полягає в створенні нового об'єкта
    }
}
```

Спосіб 2. Для створення візуального додатку можна скористатися засобами IDE Eclipse. Для цього створюємо новий проект, в ньому створюємо новий клас з категорії Other… Вибираємо пункт JFrame (можна скористатися також JDialog або Application Window).

Рисунок 1. Створення фрейму в Eclipse



Після створення класу отримаємо заготовку коду для опису фрейму. Слід звернути увагу, що в цьому випадку Eclipse створює вікно класу з двома вкладками: Source та Design (див. рис 3.) За допомогою них можна переходити від візуального конструктора вікна до редагування коду класу і навпаки. Усю роботу з додавання в код класу компонентів розміщених у вікні за допомогою конструктора IDE бере на себе.



Далі переходимо у режим конструктора. Вибираємо менеджер компоновки (Layout) з абсолютним позиціюванням (Absolute layout). Розташовуємо на фреймі необхідні компоненти (див. рис. 1): три текстові поля JTextField, три підписи JLabel та кнопку JButton. Вирівнювання та розміри елементів можна задавати візуально за допомогою перетягування мишею, або у вікні властивостей (Properties). Для текстового поля, в якому виводитиметься сила струму задаємо значення false для властивості editable. Для фрейму можна задати заголовок, наприклад, «Закон Ома», він визначається властивістю title.

Далі створюємо обробник події натискання на кнопку. Для цього можна перемкнути вікно Properties в режим керування подіями кнопкою Show events. Виділивши кнопку JButton вибираємо подію Action, у стовпці навпроти пункту performed двічі клацаємо лівою клавішею миші. Внаслідок цього Eclipse реєструє обробник натискання кнопки у вигляді анонімного класу і перемикається в режим редагування коду цього класу. Далі слід вписати код реакції програми на натискання кнопки, наприклад:

```
button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
            double r = Double.parseDouble(textField_1.getText());
            double u = Double.parseDouble(textField.getText());
            double i = u/r;
            textField_2.setText(Double.toString(i));}
        });
```

Залишається зберегти проект та для перевірки запустити фрейм на виконання.