Програмування Java.

Лабораторна робота №6

Лабораторна робота №6 присвячена побудові додатків з використанням графічного інтерфейсу бібліотек java.awt та javax.swing. По даній темі з мережі Інтернет можна рекомендувати наступні посилання:

http://www.ibm.com/developerworks/ru/edu/j-dw-java-intswing-i.html;

http://www.mexmat.sgu.ru/sites/chairs/prinf/materials/java/lesson8.htm;

http://www.uic.rsu.ru/doc/programming/java/TIJ2e.ru/Chapter13.html;

Графічні інструменти в мові Java реалізовані за допомогою двох пакетів:

* AWT (для доступу завантажується пакет java.awt) містить набір класів, що дозволяють виконувати графічні операції і створювати елементи управління;
* Swing (для доступу завантажується пакет javax.swing) містить нові класи, в основному аналогічні AWT. До іменам класів додається J (JButton, JLabel і ін).

На даний момент основні класи для побудови візуальних інтерфейсів містяться в пакеті Swing. З пакета AWT використовуються класи для обробки повідомлень. Найпростіше графічне додаток приведено нижче.

**import javax.swing. \*;**

**public final class HelloWorld implements Runnable {**

**public static void main (String [] args) {**

**// Swing має власний керуючий потік (т.зв. dispatching thread),**

**// який працює паралельно з основним (в якому виконується main ())**

**// потоком. Якщо основний потік закінчить роботу (метод main завершиться),**

**// потік відповідає за роботу Swing-інтерфейсу може продовжувати свою роботу.**

**// І навіть якщо користувач закрив всі вікна, програма продовжить свою роботу**

**// (до тих пір, поки живий даний потік). Починаючи з Java 6, коли все**

**// компоненти знищені, керуючий потік зупиняється автоматично.**

**// Запускаємо весь код, який працює в керуючому потоці, навіть ініціалізацію:**

**SwingUtilities.invokeLater (new HelloWorld ());**

**}**

**public void run () {**

**// Створюємо вікно з заголовком "Hello, World!"**

**Frame f = new JFrame ( "Hello, World!");**

**// Раніше практикувалося наступне: створювався listener і реєструвався**

**// на екземплярі головного вікна, який реагував на windowClosing ()**

**// примусової зупинкою віртуальної машини викликом System.exit ()**

**// Тепер же є більш "правильний" спосіб задавши реакцію на закриття вікна.**

**// Даний спосіб знищує поточне вікно, але не зупиняє додаток. тим**

**// самим додаток буде працювати поки не будуть закриті всі вікна.**

**f.setDefaultCloseOperation (JFrame.DISPOSE\_ON\_CLOSE);**

**// однак можна задати і так:**

**// f.setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);**

**// Додаємо на панель вікна нередагованого компоненту з текстом.**

**// f.getContentPane().Add(new JLabel( "Hello, World!")); - Старий стиль**

**f.add (new JLabel ( "Hello World"));**

**// pack () "упаковує" вікно до оптимального розміру**

**// всіх розташованих в ньому компонентів.**

**f.pack ();**

**// Показати вікно**

**f.setVisible (true);**

**}**

**}**

Технологія Swing надає механізми для управління такими аспектами уявлення:

* Клавіатура (Swing надає спосіб перехоплення призначеного для користувача введення);
* Кольори (Swing надає спосіб міняти кольори, які ви бачите на екрані);
* Текстове поле для введення (Swing надає текстові компоненти для обробки всіх повсякденних завдань).

**JComponent**

Базовим класом всій бібліотеки візуальних компонентів Swing є JComponent. Це суперклас інших візуальних компонентів. Він є абстрактним класом, тому в дійсності ви не можете створити JComponent, але він містить сотні функцій, які кожен компонент Swing може використовувати як результат ієрархії класів. Клас JComponent забезпечує інфраструктуру фарбування для всіх компонентів, він знає, як обробляти всі натискання клавіш на клавіатурі, його підкласи, отже, повинні тільки прослуховувати певні клавіші. Клас JComponent також містить метод add (), який дозволяє додати інші об'єкти класу JComponent, так можна додати будь-який Swing-компонент до будь-якого іншого для створення вкладених компонентів (наприклад, JPanel, що містить JButton, або навіть більш вигадливі комбінації, наприклад JMenu, що містить JButton ).

**JLabel**

Найпростішим і в той же час основним візуальним компонентом в бібліотеці Swing є JLabel, або «мітка». До методів цього класу відноситься установка тексту, зображення, вирівнювання та інших компонентів, які описує мітка:

get / setText () - отримати / встановити текст в мітці;

get / setIcon () - отримати / встановити зображення в мітці;

get / setHorizontalAlignment - отримати / встановити горизонтальну позицію тексту;

get / setVerticalAlignment () - отримати / встановити вертикальну позицію тексту;

get / setDisplayedMnemonic () - отримати / встановити мнемоніку (підкреслений символ) для мітки;

get / setLabelFor () - отримати / встановити компонент, до якого приєднана дана мітка; коли користувач натискає комбінацію клавіш Alt + мнемоніка, фокус переміщається на вказаний компонент.

**JButton**

Основним активним компонентом в Swing є JButton

Методи, які використовуються для зміни властивостей JButton, аналогічні методам JLabel (ви виявите, що вони аналогічні для більшості Swing-компонентів). Вони управляють текстом, зображеннями та орієнтацією:

get / setText () - отримати / встановити текст в кнопці;

get / setIcon () - отримати / встановити зображення в кнопці;

get / setHorizontalAlignment () - отримати / встановити горизонтальну позицію тексту;

get / setVerticalAlignment () - отримати / встановити вертикальну позицію тексту;

get / setDisplayedMnenomic () - отримати / встановити мнемоніку (підкреслений символ), яка в комбінації з кнопкою Alt викликає натискання кнопки.

**JFrame**

Клас JFrame є контейнером, що дозволяє додавати до себе інші компоненти для їх організації та надання користувачеві.

JFrame виступає в якості моста між незалежними від конкретної операційної системи Swing-частинами і реальної операційною системою, на якій вони працюють. JFrame реєструється як вікно і таким чином отримує багато властивостей вікна операційної системи: мінімізація / максимізація, зміна розмірів і переміщення. Хоча виконанні лабораторної роботи досить вважати JFrame палітрою, на якій ви розміщуєте компоненти. Перерахуємо деякі з методів, які ви можете викликати в JFrame для зміни його властивостей:

get / setTitle () - отримати / встановити заголовок фрейму;

get / setState () - отримати / встановити стан фрейма (мінімізувати, максимізувати і т.д.);

is / setVisible () - отримати / встановити видимість фрейма, іншими словами, відображення на екрані;

get / setLocation () - отримати / встановити місце розташування у вікні, де фрейм повинен з'явитися;

get / setSize () - отримати / встановити розмір фрейму;

add () - додати компоненти до кадру.

**Схеми, моделі та події**

При побудові візуальних додатків в Java не можна просто випадково розмістити їх на екрані і очікувати від них негайної роботи. Компоненти необхідно розмістити в певні місця, реагувати на взаємодію з ними, оновлювати їх на основі цієї взаємодії і заповнювати даними. Для ефективної роботи з візуальними компонентами необхідна установка наступних трьох архітектурних складових Swing.

1. Схеми (layout). Swing містить безліч схем, які представляють собою класи, керуючі розміщенням компонентів в додатку і тим, що повинно статися з ними при зміні розмірів вікна програми або при видаленні або додаванні компонентів.

2. Події (event). Програма повинна реагувати на натискання клавіш, натискання кнопки миші і на все інше, що користувач може зробити.

3. Моделі (model). Для більш просунутих компонентів (списки, таблиці, дерева) і навіть для деяких більш простих, наприклад, JComboBox, моделі - це найефективніший спосіб роботи з даними. Вони видаляють більшу частину роботи по обробці даних з самого компонента (згадайте обговорення MVC) і надають оболонку для загальних об'єктних класів даних (наприклад, Vector і ArrayList).

Особливу увагу, в зв'язку з необхідністю зображення динамічних сцен на візуальних компонентах необхідно приділити класу Graphics2D.

Приклад рішення одного із завдань на лабораторну роботу №4

Завдання: Відобразити обертання трикутника навколо свого центра ваги

Рішення:

import java.awt. \*;

import java.awt.event. \*;

import java.awt.geom. \*

Import javax.swing. \*;

public class Main

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

JFrame fr = new JFrame ( "Обертання трикутника навколо свого центра ваги");

fr.setPreferredSize(**new** Dimension(300,300));

**final** JPanel pan= **new** JPanel();

fr.add(pan);

fr.setVisible(**true**);

fr.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

fr.pack();

Timer tm= **new** Timer(500, **new** ActionListener(){

**int** i=0;

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

Graphics2D gr=(Graphics2D)pan.getRootPane().getGraphics();

pan.update(gr);

GeneralPath path=**new** GeneralPath();

path.append(**new** Polygon(**new** **int** []{60,-80,50},**new** **int**[]{-60,-50,40},3),**true**);

**int** x=(60-80+50)/3,y=(-60-50+40)/3;

gr.translate(150, 150);

AffineTransform tranforms = AffineTransform.*getRotateInstance*((i++)\*0.07, x, y);

gr.transform(tranforms);

gr.draw(path);

}});

tm.start();

}

}

Індивідуальних завдань для виконання

лабораторної роботи №4

Завдання 1.

1. Задати рух по екрану рядків (одна за одною) з масиву рядків. Напрямок руху по фрейму і значення кожного рядка вибирається випадковим чином.

2. Задати рух окружності з фрейму так, щоб при торканні кордону окружність відбивалася від неї з ефектом пружного стиснення.

3. Зобразити в фреймі наближається здалеку куля, що віддаляється куля. Куля повинна рухатися з постійною швидкістю.

4. Зобразити у вікні програми відрізок, що обертається в площині екрану навколо однієї зі своїх кінцевих точок. Колір прямий повинен змінюватися при переході від одного положення до другому.

5. Зобразити у вікні програми відрізок, що обертається в площині фрейму навколо точки, що рухається по відрізку.

6. Зобразити чотирикутник, що обертається в площині фрейму навколо свого центру ваги.

7. Створити фрейм з областю для малювання «пером». Створити меню для вибору кольору і товщини лінії.

8. Скласти програму для управління швидкістю руху точки по фрейму. Одна кнопка збільшує швидкість, інша - зменшує. Кожне клацання змінює швидкість на певну величину.

9. Зобразити у вікні гармонійні коливання точки вздовж деякого горизонтального відрізка. Якщо довжина відрізка дорівнює q, то відстань від точки до лівого кінця в момент часу t можна вважати рівним q (1 + cos (wt)) / 2, де w - деяка константа. Передбачити поля для введення зазначених величин і кнопку для зупинки і старту процесу.

10. Створити фрейм з рядком рухається по діагоналі. При досягненні меж фрейму всі символи рядка випадковим чином змінюють регістр. При цьому шрифт змінюється на шрифт, вибраний з списку.

11. Створити фрейму з рядком рухається горизонтально, відбиваючись від границь фрейму і змінюючи при цьому свій колір, на колір обраний зі спадного списку.

12. Промоделювати обертання супутника навколо планети по еліптичній орбіті. Коли ховається за планетою - супутнику не видно.

13. Промоделювати аналоговий годинник (зі стрілками) з кнопками для збільшення / зменшення часу на годину / хвилину.

Завдання 2.

Для програми розробленої у ході виконання лабораторної роботи №1 (частина 3) розробити візуальний інтерфейс засобами бібліотеки Swing. У програмі використати такі компоненти JFrame, JPanel, JButton, JLable, JTextField, JTable. Передбачити читання вхідних даних з файлу.

Програма повинна включати обробку виключних ситуацій. Обробити 3 типи виключень: 2 стандартних і 1 власне. Зі стандартних виключень можна взяти виключення пов’язане з відкриттям файлу і невірним форматом вхідних даних. Для власного виключення потрібно розробити клас наслідуваний від класу ArithmeticException. В програмі передбачити генерацію власного виключення при певних вхідних даних (придумати самостійно).