ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ “МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ”.

Кафедра МКиИТ

Лабораторная работа №6 по дисциплине: ”Технологии программирования ”

**“**ЗаданиеNo6: многоядерный исследователь/генератор фракталов **”.**

Выполнила: студентка группы БСТ1602

Лушина Ольга

Проверил: М. Г. Городничев

Москва 2018

**Содержание:**

1. Цели и задачи. 3
2. Анализ предметной области и выбор инструментария. 3
3. Функции и их объяснение. 3
4. Выводы. 9
5. Цели и задачи: изменим программу так, чтобы она использовала один или несколько фоновых ядер для вычисления фрактала.
6. Анализ предметной области и выбор инструментария: в данной лабораторной работе я использовала бесплатный пакет Jdk.
7. Функции и их объяснение:
8. *Файл FractalExplorer.java.*

*Создаём подкласс FractalWorker, который предназначен для вычислений значений цвета одной строки или ряда.*

*Метод doInBackground () отвечает за выполнение долгосрочной задачи.*

*Метод done() вызывается, когда фоновая задача завершена.*

*Изменяем функцию drawFractal , которая запустит фоновый поток и запустит задачу в фоновом режиме.*

|  |
| --- |
| import java.awt.geom.Rectangle2D; |
|  |

|  |
| --- |
| import java.awt.BorderLayout; |
|  |

|  |
| --- |
| import javax.imageio.ImageIO; |
|  |

|  |
| --- |
| import java.awt.\*; |
|  |

|  |
| --- |
| import java.awt.event.\*; |
|  |

|  |
| --- |
| import javax.swing.\*; |
|  |

|  |
| --- |
| import java.awt.event.\*; |
|  |

|  |
| --- |
| import java.io.\*; |
|  |

|  |
| --- |
| import javax.swing.filechooser.FileNameExtensionFilter; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| public class FractalExplorer |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| public static void main(String[] args) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| FractalExplorer MyExplorer = new FractalExplorer(500); |
|  |

|  |
| --- |
| MyExplorer.createAndShowGUI(); |
|  |

|  |
| --- |
| MyExplorer.drawFractal(); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| private int MyDispalySize; |
|  |

|  |
| --- |
| private int MySize; |
|  |

|  |
| --- |
| private int RowsRem; |
|  |

|  |
| --- |
| private JImageDisplay MyImage; |
|  |

|  |
| --- |
| private FractalGenerator MyGenerator; |
|  |

|  |
| --- |
| private FractalGenerator MyGeneratorT; |
|  |

|  |
| --- |
| private FractalGenerator MyGeneratorM; |
|  |

|  |
| --- |
| private FractalGenerator MyGeneratorB; |
|  |

|  |
| --- |
| private Rectangle2D.Double MyRange= new Rectangle2D.Double(0,0,0,0); |
|  |

|  |
| --- |
| private JComboBox Box =new JComboBox(); |
|  |

|  |
| --- |
| private Button B=new Button("Reset"); |
|  |

|  |
| --- |
| private Button S=new Button("Save"); |
|  |

|  |
| --- |
| FractalExplorer(int NewSize) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MySize=NewSize; |
|  |

|  |
| --- |
| MyImage = new JImageDisplay(NewSize,NewSize); |
|  |

|  |
| --- |
| MyGenerator= new Tricorn(); |
|  |

|  |
| --- |
| MyGeneratorT= new Tricorn(); |
|  |

|  |
| --- |
| MyGeneratorM= new Mandelbrot(); |
|  |

|  |
| --- |
| MyGeneratorB= new BurningShip(); |
|  |

|  |
| --- |
| MyGenerator.getInitialRange(MyRange); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| private void enableUI(boolean val) { |
|  |

|  |
| --- |
| B.setEnabled(val); |
|  |

|  |
| --- |
| S.setEnabled(val); |
|  |

|  |
| --- |
| Box.setEnabled(val); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| public void createAndShowGUI() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| JFrame MyFrame =new JFrame(); |
|  |

|  |
| --- |
| MyFrame.setLayout(new BorderLayout()); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| JLabel MyLable=new JLabel("FRACTAL"); |
|  |

|  |
| --- |
| JPanel MyPanel=new JPanel(); |
|  |

|  |
| --- |
| JPanel MyPanel2=new JPanel(); |
|  |

|  |
| --- |
| MyPanel.add(MyLable); |
|  |

|  |
| --- |
| MyPanel.add(Box); |
|  |

|  |
| --- |
| MyFrame.add(MyPanel, BorderLayout.NORTH); |
|  |

|  |
| --- |
| Box.addItem(MyGeneratorT); |
|  |

|  |
| --- |
| Box.addItem(MyGeneratorM); |
|  |

|  |
| --- |
| Box.addItem(MyGeneratorB); |
|  |

|  |
| --- |
| MyPanel2.add(B); |
|  |

|  |
| --- |
| MyPanel2.add(S); |
|  |

|  |
| --- |
| MyFrame.add(MyPanel2, BorderLayout.SOUTH); |
|  |

|  |
| --- |
| MyFrame.add(MyImage, BorderLayout.CENTER); |
|  |

|  |
| --- |
| MyFrame.setDefaultCloseOperation(MyFrame.EXIT\_ON\_CLOSE); |
|  |

|  |
| --- |
| MyFrame.pack(); |
|  |

|  |
| --- |
| MyFrame.setVisible(true); |
|  |

|  |
| --- |
| MyEventListener handler2=new MyEventListener(); |
|  |

|  |
| --- |
| MyFrame.setResizable(false); |
|  |

|  |
| --- |
| MouseMyAdapter handler=new MouseMyAdapter(); |
|  |

|  |
| --- |
| MyImage.addMouseListener(handler); |
|  |

|  |
| --- |
| B.addActionListener(handler2); |
|  |

|  |
| --- |
| S.addActionListener(handler2); |
|  |

|  |
| --- |
| Box.addActionListener(handler2); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| public void drawFractal() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| enableUI(false); |
|  |

|  |
| --- |
| RowsRem=MySize; |
|  |

|  |
| --- |
| for(int y=0;y<MySize;y++) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| FractalWorker MyWorker = new FractalWorker(y); |
|  |

|  |
| --- |
| MyWorker.execute(); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| public class MouseMyAdapter extends MouseAdapter |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| public void mouseClicked(MouseEvent e) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if (RowsRem!=0) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| return; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| MyGenerator.recenterAndZoomRange(MyRange,FractalGenerator.getCoord(MyRange.x,MyRange.x+MyRange.width,MySize,e.getX()),FractalGenerator.getCoord(MyRange.y,MyRange.y+MyRange.height,MySize,e.getY()),0.2); |
|  |

|  |
| --- |
| drawFractal(); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| public class MyEventListener implements ActionListener |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| public void actionPerformed(ActionEvent e) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if(e.getSource().equals(B)) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MyGenerator.getInitialRange(MyRange); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| if(e.getSource().equals(S)) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| JFileChooser ChooseFile = new JFileChooser(); |
|  |

|  |
| --- |
| FileNameExtensionFilter FileFilter = new FileNameExtensionFilter("PNG Images","png"); |
|  |

|  |
| --- |
| ChooseFile.setFileFilter(FileFilter); |
|  |

|  |
| --- |
| ChooseFile.setAcceptAllFileFilterUsed(false); |
|  |

|  |
| --- |
| if(ChooseFile.showSaveDialog(null)==JFileChooser.APPROVE\_OPTION) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| try |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| File file = ChooseFile.getSelectedFile(); |
|  |

|  |
| --- |
| String PathM = file.getPath(); |
|  |

|  |
| --- |
| file=new File(PathM+ ".png"); |
|  |

|  |
| --- |
| ImageIO.write(MyImage.getImage(),"png",file); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| catch(IOException err) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| JOptionPane.showMessageDialog(null,"error"); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| if(e.getSource().equals(Box)) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| String H=Box.getSelectedItem().toString(); |
|  |

|  |
| --- |
| if(H.equals("Mandelbrot")) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MyGenerator=MyGeneratorM; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| if(H.equals("Tricorn")) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MyGenerator=MyGeneratorT; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| if(H.equals("BurningShip")) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MyGenerator=MyGeneratorB; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| MyGenerator.getInitialRange(MyRange); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| drawFractal(); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| public class FractalWorker extends SwingWorker<Object,Object> |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| int y2; |
|  |

|  |
| --- |
| int[] S; |
|  |

|  |
| --- |
| public FractalWorker(int a) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| y2=a; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| public Object doInBackground() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| S= new int[MySize]; |
|  |

|  |
| --- |
| double Y = FractalGenerator.getCoord(MyRange.y,MyRange.y+MyRange.height,MySize,y2); |
|  |

|  |
| --- |
| for(int x=0;x<MySize;x++) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| double X = FractalGenerator.getCoord(MyRange.x,MyRange.x+MyRange.width,MySize,x); |
|  |

|  |
| --- |
| int iteration = MyGenerator.numIterations(X,Y); |
|  |

|  |
| --- |
| int ColoR=0; |
|  |

|  |
| --- |
| if(iteration!=-1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| float hue = 0.7f + (float)iteration/200f; |
|  |

|  |
| --- |
| ColoR = Color.HSBtoRGB(hue,1f,1f); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| S[x]=ColoR; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| return null; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| public void done() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| for(int x=0;x<MySize;x++) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| MyImage.drawPixel(x,y2,S[x]); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| MyImage.repaint(0,0,y2,MySize,1); |
|  |

|  |
| --- |
| RowsRem--; |
|  |

|  |
| --- |
| if (RowsRem<1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| enableUI(true); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

}

1. Выводы: получилась программа, которая может рисовать фракталы с несколькими потоками и это не позволит пользователям ничего делать, пока процесс отрисовки происходит в фоновом режиме.