ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ

Ордена Трудового Красного Знамени

федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский Технический Университет Связи и Информатики» (МТУСИ)

Кафедра МКиИТ

Лабораторная работа №5

«Многоядерный исследователь/генератор фракталов»

Выполнил:

Студент 2 курса

Группы БСТ-1602

Смышляев Владислав

Вариант №15

Москва, 2018

Содержание

[1. Цель и задачи 3](#_Toc512781951)

[2. Анализ предметной области, выбор инструментария 3](#_Toc512781952)

[3. Объяснение функций 3](#_Toc512781953)

[3.1. Класс FractalWorker 3](#_Toc512781954)

[4. Выводы 4](#_Toc512781955)

1. Цель и задачи

Цель работы: Изучить способ сделать программу многопоточной.

Задачи: Написать программу рисования фрактала, которая будет использовать не один поток, а множество для ускорения работы.

1. Анализ предметной области, выбор инструментария

Для написания данной программы использоваться IDE IntelliJ Idea, так как в данной и последующих задачах требуются средства для отладки программы. Здесь также представлены средства для работы с контролем версий, но для собственного удобства коммиты будет совершаться через Visual Studio Code.

1. Объяснение функций

Так как, данная работа основа на предыдущей здесь будут представлены лишь модификации кода, которые требовались для выполнения задачи.

* 1. Класс FractalWorker

*/\*\*  
 \* Allows for multi-threading drawing of fractals  
 \*/*private class FractalWorker extends SwingWorker<Object,Object> {  
 private int y\_coord;  
 private int[] rgb\_values;  
  
 */\*\*  
 \* Constructs the worker  
 \** ***@param*** *y - row to draw on  
 \*/* FractalWorker(int y){  
 y\_coord = y;  
 }  
  
 @Override  
 protected Object doInBackground() {  
 rgb\_values = new int[dispSize];  
 for (int i=0;i<dispSize;i++)  
 {  
 //Map display coordinates to fractal coordinates  
 double xCoord = FractalGenerator.*getCoord*(plane\_range.x,plane\_range.x + plane\_range.width,dispSize,i);  
 double yCoord = FractalGenerator.*getCoord*(plane\_range.y,plane\_range.y + plane\_range.height,dispSize,y\_coord);  
  
 //Get the iterations amount  
 int iters = generator.numIterations(xCoord,yCoord);  
  
 //Default pixel color to black  
 int px\_color = 0;  
  
 //Check if we are in the limit of our fractal  
 if (iters != -1)  
 {  
 //Calcualte the pixel color  
 float hue = 0.7f + (float) iters / 200f;  
 px\_color = Color.*HSBtoRGB*(hue,1f,1f);  
 }  
 //Save the pixel color into an array  
 rgb\_values[i] = px\_color;  
 }  
 return null;  
 }  
  
 @Override  
 protected void done() {  
 super.done();  
 //Draw the row of pixels  
 for(int i=0;i<dispSize;i++){  
 display.drawPixel(i,y\_coord,rgb\_values[i]);  
 }  
 //Repaint the screen  
 display.repaint(0,0,y\_coord,dispSize,1);  
  
 if (--remaining\_rows == 0){  
 enableUI(true);  
 }  
 }  
}

Единственным новым классом, который был добавлен в данной работе, это внутренний класс FractalWorker, с помощью которого будет обеспечиваться многопоточность программы. Для этого наследуется класс SwingWorker, который за нас обеспечит работу данного класса на потоках отличных от основного. Здесь стоит только переписать функции doInBackground() и done(), описывающие, что необходимо сделать потоку. Данный класс рисует фрактал вертикально, попиксельно.

Таким способом добивается ускоренная прорисовка фрактала.

1. Выводы

Таким образом, был изучен способ создания многопоточной программы.