Лабораторная работа №5. Выбор и сохранение фракталов

В данной лабораторной работе генератор фракталов будет расширен двумя новыми функциями. Во-первых, вы добавите поддержку нескольких фракталов и реализуете возможность выбирать нужный фрактал из выпадающего списка. Во-вторых, вы добавите поддержку сохранения текущего изображения в файл. Ниже приведен скриншот, где продемонстрировано, как будет выглядеть новая программа

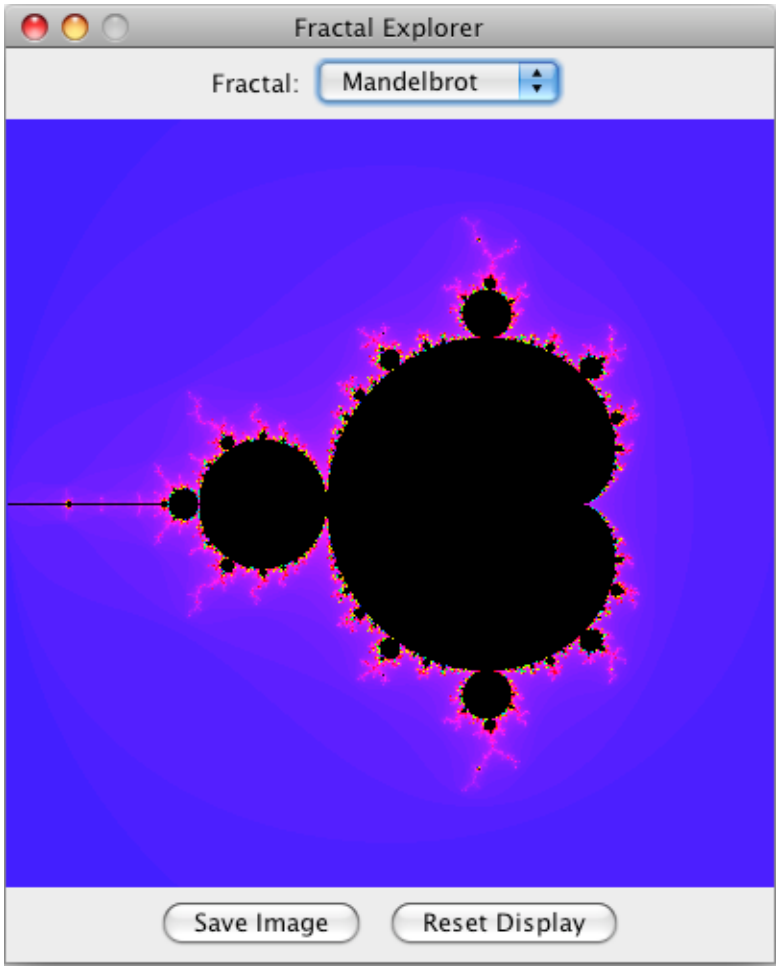

Рисунок 5.5. Графический интерфейс нового приложения

Верхняя панель генератора фракталов включает в себя 2 виджета, позволяющих пользователю выбирать фрактал, а нижняя панель включает в себя кнопку "Save Image", которая сохраняет текущее изображения фрактала.

Так как теперь будет несколько источников событий (action-event sources), вы сможете попрактиковаться в обработке всех источников с использованием одного метода ActionListener в вашем классе.

Поддержка нескольких фракталов

Так как в реализацию была введена абстракция FractalGenerator, добавление нескольких фракталов не будет проблемой. В данной лабораторной работе вы добавите поддержку нескольких фракталов, и пользователь сможет выбирать между ними, используя *combo-box*. Программный интерфейс Swing (Swing API) предоставляет *combo-box* через класс javax.swing.JComboBox, а также запускает ActionEvents при выборе новогоэлемента. Необходимо сделать:

- Создать 2 новые реализации FractalGenerator

Первым будет фрактал tricorn, который должен находиться в файле Tricorn.java. Для этого нужно создать подкласс FractalGenerator и реализация будет почти идентична фракталу Мандельброта, кроме двух изменений. Вы даже можете скопировать исходный код фрактала Мандельберта и просто внести следующие изменения:

- Уравнение имеет вид $z_n = z_{n-1}^2 + c$. Единственное отличие только в том, что используется комплексное сопряжение z_{n-1} на каждой итерации.
- Начальный диапазон для трехцветного фрактала должен быть от *(-2, -2)* до *(2, 2)*.

Второй фрактал, который необходимо реализовать - это фрактал «Burning Ship», который в реальности не похож на пылающий корабль. Данный фрактал имеет следующие свойства:

- Уравнение имеет вид $z_n = (|\mathrm{Re}(z_{n-1})| + i\ |\mathrm{Im}(z_{n-1})|)^2 + c$. Другими словами, вы берете абсолютное значение каждого компонента z_{n-1} на каждой итерации.
- Начальный диапазон для данного фрактала должен быть от *(-2, -2.5)* до *(2, 1.5)*.

- Combo-boxe в Swing может управлять коллекцией объектов, но объекты должны предоставлять метод toString(). Убедитесь, что в каждой реализации фракталов tcnm метод toString(), который возвращает имя, например «Mandelbrot», «Tricorn» и «Burning Ship».

- Настроить JComboBox в вашем пользовательском интерфейсе можно с использованием конструктора без параметров, а затем использовать метод addItem(Object) для того, чтобы добавить реализации вашего генератора фракталов. Как указывалось в предыдущем шаге, выпадающий список будет использовать метод toString () в ваших реализациях для отображения генераторов в выпадающем списке.

Необходимо будет также добавить объект label в разрабатываемый пользовательский интерфейс перед выпадающим списком, в качестве пояснения к выпадающему списку. Это можно сделать, создав новый объект Jpanel и добавив в него объекты JLabel и JcomboBox, а затем разместить панель на позиции NORTH на вашем макете окна.

И наконец, необходимо добавить поддержку выпадающего списка в реализацию ActionListener. В случае, если событие поступило от выпадающего списка, вы можете извлечь выбранный элемент из виджета и установить его в качестве текущего генератора фракталов. (Используйте метод getSelectedItem()) При этом не забудьте сбросить начальный диапазон и перерисовать фрактал!

Ниже приведены изображения фракталов «Tricorn» и «Burning Ship» для проверки правильности работы алгоритма

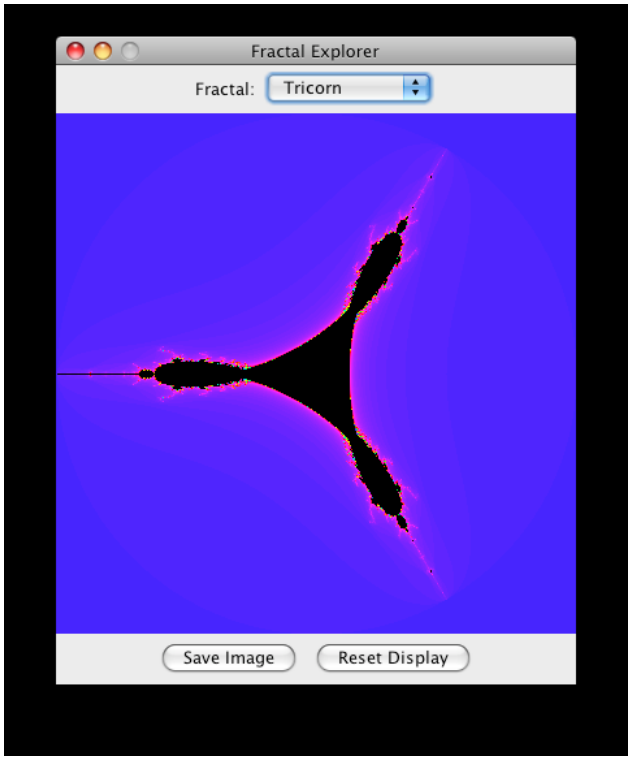

Рисунок 5.6. Фрактал «Tricorn»

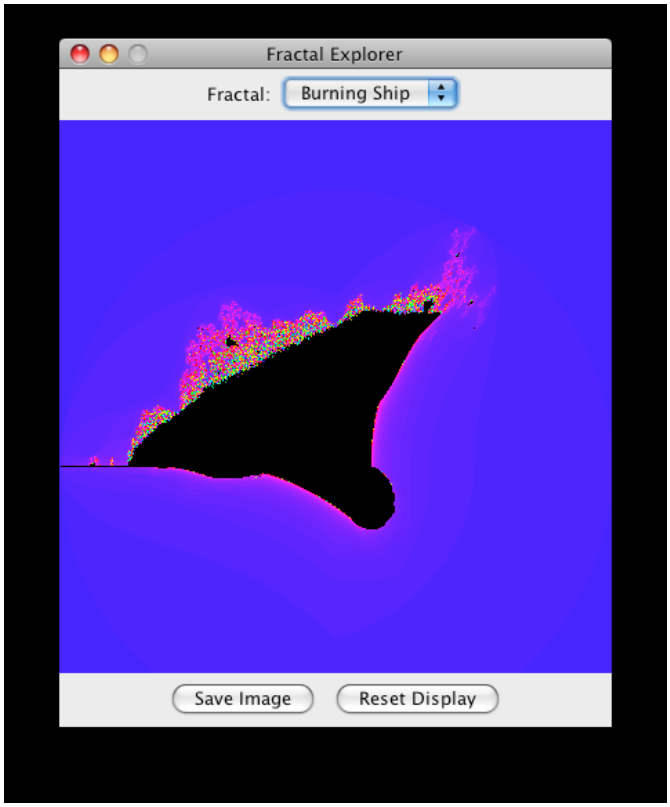

Рисунок 5.7. Фрактал «Burning Ship»

Сохранения изображения Фрактала

Следующая ваша задача - сохранение текущего изображения фрактала на диск. Java API предоставляет несколько инструментов для реализации данной задачи.

- Во-первых, вам нужно добавить кнопку «Save Image» в ваше окно. Для этого вы можете добавить обе кнопки «Save Image» и «Reset» в новую Jpanel, а затем разместить эту панель в SOUTH части окна.

События от кнопки «Save Image» также должны обрабатываться реализацией ActionListener. Назначьте кнопкам «Save Image» и «Reset» свои значения команд (например, «save» и «reset») для того, чтобы обработчик событий мог отличить события от этих двух разных кнопок.

- В обработчике кнопки «Save Image» вам необходимо реализовать возможность указания пользователем, в какой файл он будет сохранять изображение. Это можно сделать с помощью класса javax.swing.JFileChooser. Указанный класс предоставляет метод showSaveDialog(), который открывает диалоговое окно «Save file», позволяя тем самым пользователю выбрать директорию для сохранения. Метод принимает графический компонент, который является родительским элементом для диалогового окна с выбором файла, что позволяет центрированию окна с выбором относительно его родителя. В качестве родителя используйте окно приложения.

Как вы могли заметить, данный метод возвращает значение типа int, которое указывает результат операции выбора файла. Если метод возвращает значение JfileChooser.APPROVE_OPTION, тогда можно продолжить операцию сохранения файлов, в противном случае, пользователь отменил операцию, поэтому закончите данную обработку события без сохранения. Если пользователь выбрал директорию для сохранения файла, вы можете ее узнать, используя метод getSelectedFile(), который возвращает объект типа File.

- Также необходимо настроить средство выбора файлов, чтобы сохранять изображения только в формате PNG, на данном этапе вы будете работать только с данным форматом. вы сможете это настроить с помощью

javax.swing.filechooser.FileNameExtensionFilter, как это продемонстрировано ниже:

```
JFileChooser chooser = new JfileChooser();
FileFilter filter = new FileNameExtensionFilter("PNG Images", "png");
chooser.setFileFilter(filter);
chooser.setAcceptAllFileFilterUsed(false);
```

Последняя строка гарантирует, что средство выбора не разрешит пользователю использование отличных от png форматов.

- Если пользователь успешно выбрал файл, следующим шагом является сохранения изображения фрактала на диск! Для данного рода задач Java включает в себя необходимую функциональность. Класс javax.imageio.ImageIO обеспечивает простые операции загрузки и сохранения изображения. Вы можете использовать метод write(RenderedImage im, String formatName, File output). Параметр formatName будет содержать значение «png”. Тип «RenderedImage» - это просто экземпляр BufferedImage из вашего компонента JimageDisplay. (Используйте для него тип доступа public)

Метод write() может вызвать исключение, поэтому вам необходимо заключить этот вызов в блок try/catch и обработать возможную ошибку. Блок catch должен проинформировать пользователя об ошибке через диалоговое окно. Swing предоставляет класс javax.swing.JoptionPane для того, чтобы упростить процесс создания информационных диалоговых окон или окон, где нужно выбрать да/нет. Для этого вы можете использовать статический метод JoptionPane.showMessageDialog(Component parent, Object message, String title, int messageType), где messageType у вас будет JOptionPane.ERROR_MESSAGE. В сообщении об ошибке вы можете использовать возвращаемое значение метода getMessage(), а заголовком окна может быть, например, «Cannot Save Image». Родительским компонентом будет окно для того, чтобы диалоговое окно с сообщением об ошибке выводилось относительно центра окна.

После того, как вы закончите реализацию этих функций, запустите. Теперь вы сможете исследовать различные фракталы, а также вы сможете

сохранять их на диск. Вы также можете проверить приложение на вывод сообщений об ошибках, попробуйте сохранить изображение в файл, который уже существует, но доступен только для чтения. Или вы можете попробовать сохранить файл с именем, которое является каталогом в целевой папке.