Jason Mejia
Professor Auda
CSC221

11/14/2019

Project #3 Report

Objective:

In this project it requires to make a pie chart out of the most frequent letters in a text file. And we are using the text file emma.txt

Solution:

I started working on the code that reads the text file first. Then proceeded to the code that creates the pie chart and then to the Histogram letters. The hashmap values were: letter frequencies, letter probabilities, and letter angles for the pie chart. For the most frequent and the least frequent they are in sorted order characters. To create the pie chart we are using a similar method form the previous project. We are using the method fillArc() from javafx.

Code:

ReadTextFile.java

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

    public class ReadTextFile {
       private static Scanner input;
        public static void openFile(String file) {
           try {
               input = new Scanner(new File(file));
           }
           catch(FileNotFoundException e) {
               e.printStackTrace();
           }
       }
       public static String readFile() {
           String s = "";
           while(input.hasNext()) {
               s += input.nextLine();
               //System.out.println(s);
           }
           return s;
       }
       public static void closeFile() {
           if (input != null) {
               input.close();
           }
       }
   }
```

HistogramLetters.java

```
import java.util.*;

  public class HistgramLetters {

       HashMap<Character, Integer> letters_frequencies;
       HashMap<Character, Double> letters_probabilities, letters_angles;
       ArrayList<Character> nSorted;
      double char_count;


      public HistgramLetters(String file) {
          setLetters_frequencies(file);
          this.letters_probabilities = null;
          this.letters_angles = null;
          this.nSorted = null;
      }

      public HashMap getLetter_frequencies () {
          return letters_frequencies;
      }
      public HashMap getLetter_probabilites () {
          return letters_probabilities;
      }
      public HashMap getLetters_angles () {
          return letters_angles;
      }
      public ArrayList getNSorted () {
          return nSorted;
      }

      public void setLetters_frequencies(String file) {

          ReadTextFile.openFile(file);
          String temp = ReadTextFile.readFile();
          String all_letters = temp.replaceAll("[^a-zA-Z]", "").toLowerCase();

          HashMap<Character, Integer> letter_frequencies = new HashMap<Character,
Integer>();

          for (int i = 0; i < all_letters.length(); i++) {
              char c = all_letters.charAt(i);
              var val = letter_frequencies.get(c);
              if (val != null) {                      //If letter already exist,
increment value
                  letter_frequencies.put(c, val + 1);
              }
              else {
                  letter_frequencies.put(c, 1);
              }
              char_count = i;
          }
          this.letters_frequencies = letter_frequencies;
      }
```

```
      public void setLetters_probabilities(HashMap<Character, Integer>
letters_frequencies) {
          HashMap<Character, Double> letters_probabilities = new HashMap<Character,
Double>();
          letters_frequencies.forEach((k, v) -> letters_probabilities.put(k, v /
char_count));
          this.letters_probabilities = letters_probabilities;
      }

      public void setLetters_angles(HashMap<Character, Double> letters_probabilities)
{
          HashMap<Character, Double> letters_angles = new HashMap<Character,
Double>();
          letters_probabilities.forEach((k, v) -> letters_angles.put(k, v * 360));
          this.letters_angles = letters_angles;
      }

      //Creates an arraylist that stores the characters in the order of most frequent
to the least frequent

      public void setNSorted(HashMap<Character, Integer> m) {
          List<Map.Entry<Character, Integer>> list =
                  new LinkedList<Map.Entry<Character, Integer>>(m.entrySet());

          // Sort the list
          Collections.sort(list, new Comparator<Map.Entry<Character, Integer>>() {
              public int compare(Map.Entry<Character, Integer> o1,
                                 Map.Entry<Character, Integer> o2) {
                  return (o1.getValue()).compareTo(o2.getValue());
              }
          });

          int i = 0;
          ArrayList<Character> temp = new ArrayList<Character>();
          for (Map.Entry<Character, Integer> aa : list) {
              temp.add(i, aa.getKey());
          }
          this.nSorted = temp;

      }
   }
```

PieChart.java

```
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.ArcType;
import javafx.scene.text.Text;

   import java.util.*;

   public class PieChart {

      HashMap<Character, Double> probabilities, angles;
      ArrayList<Character> n_sorted;

      public PieChart(HashMap probabilities, HashMap angles, ArrayList<Character>
n_sorted) {
          this.probabilities = probabilities;
          this.angles = angles;
          this.n_sorted = n_sorted;
      }

      public void drawChart (GraphicsContext gc, Pane root, int n, double canvasW,
double canvasH) {
          double start_angle = 0;
          double end_angle = 0;
          double total_probabilities = 0;
          double centerX = canvasW/2;
          double centerY = canvasH/2;
          double RADIUS = 300;


          ArrayList<Text> labels = new ArrayList<Text>();

          for (int i = 0; i < n; i ++) {

              Random rand = new Random();

              gc.setFill((Color.rgb(rand.nextInt(255), rand.nextInt(255),
rand.nextInt(255))));

              end_angle = angles.get(n_sorted.get(i));
              //Next lines for testing purposes
              //System.out.println("Letter: " + n_sorted.get(i) +  " Angle: " +
end_angle + " Start Angle: " +
              // start_angle + " Color: " + gc.getFill());


              gc.fillArc(canvasW / 2 - RADIUS, canvasH / 2 - RADIUS, 2*RADIUS,
2*RADIUS,
                      start_angle, end_angle, ArcType.ROUND);
```

```
            //Text constructor takes x, y coordinates, and the string
            //The X field uses the function: CenterX + (radius * cos(angle)) to
find the coordinate,
            //it's also similar for the y coordinate. I added half of the next
angle to center the text.

            labels.add(new Text(centerX + ((RADIUS+50) * (Math.cos((start_angle +
.5 * end_angle )* (Math.PI/180)))),
                    centerY + ((RADIUS+50) * -Math.sin((start_angle + .5 *
end_angle )* (Math.PI/180))),
                    n_sorted.get(i) + " , " +
                            Math.round((probabilities.get((n_sorted.get(i)))) *
1000.0)/ 1000.0));
                            //Probabilities are rounded of to thousands place

            start_angle += angles.get(n_sorted.get(i));
            total_probabilities += probabilities.get(n_sorted.get(i));

            root.getChildren().add(labels.get(i));

        }
        //If n doesn't include all letters, make extra slice for all other letters
and a text object

        Random rand = new Random();

        gc.setFill((Color.rgb(rand.nextInt(255), rand.nextInt(255),
rand.nextInt(255))));
        if (n < n_sorted.size()) {
            gc.fillArc(canvasW / 2 - RADIUS, canvasH / 2 - RADIUS, 2*RADIUS,
2*RADIUS,
                    start_angle, 360 - start_angle, ArcType.ROUND);

            labels.add(new Text(centerX + ((RADIUS+50) *
                    (Math.cos((start_angle + .5 * (360 - start_angle) )*
(Math.PI/180)))),
                    centerY + ((RADIUS+50) * -Math.sin((start_angle + .5 * (360 -
start_angle) )* (Math.PI/180))),
                    "All other letters, " + Math.round((1 - total_probabilities) *
1000.0)/ 1000.0));

            root.getChildren().add(labels.get(n));
        }


    }

    public void nInput(GraphicsContext gc, Pane root, double canvasW, double
canvasH) {


        TextField b = new TextField("Enter n");
        Button button = new Button("Enter");
```

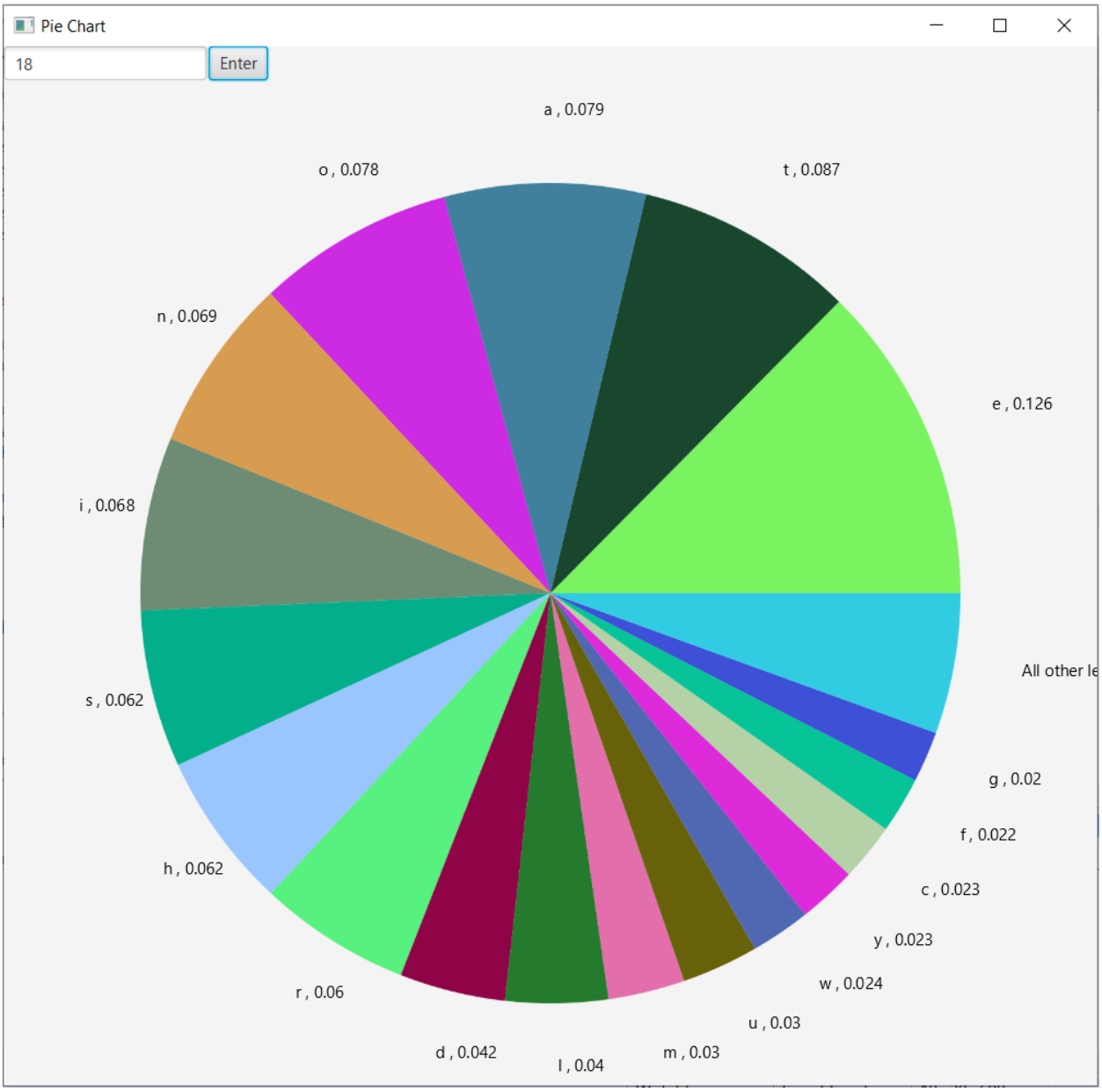
```
            button.setLayoutX(150);
            root.getChildren().add(b);

            EventHandler<ActionEvent> event = new EventHandler<ActionEvent>() {
                public void handle(ActionEvent e)
                {
                   drawChart(gc, root, Integer.parseInt(b.getText()), canvasW,
canvasH);
                }
            };
            button.setOnAction(event);
            root.getChildren().add(button);

    }


  }
```

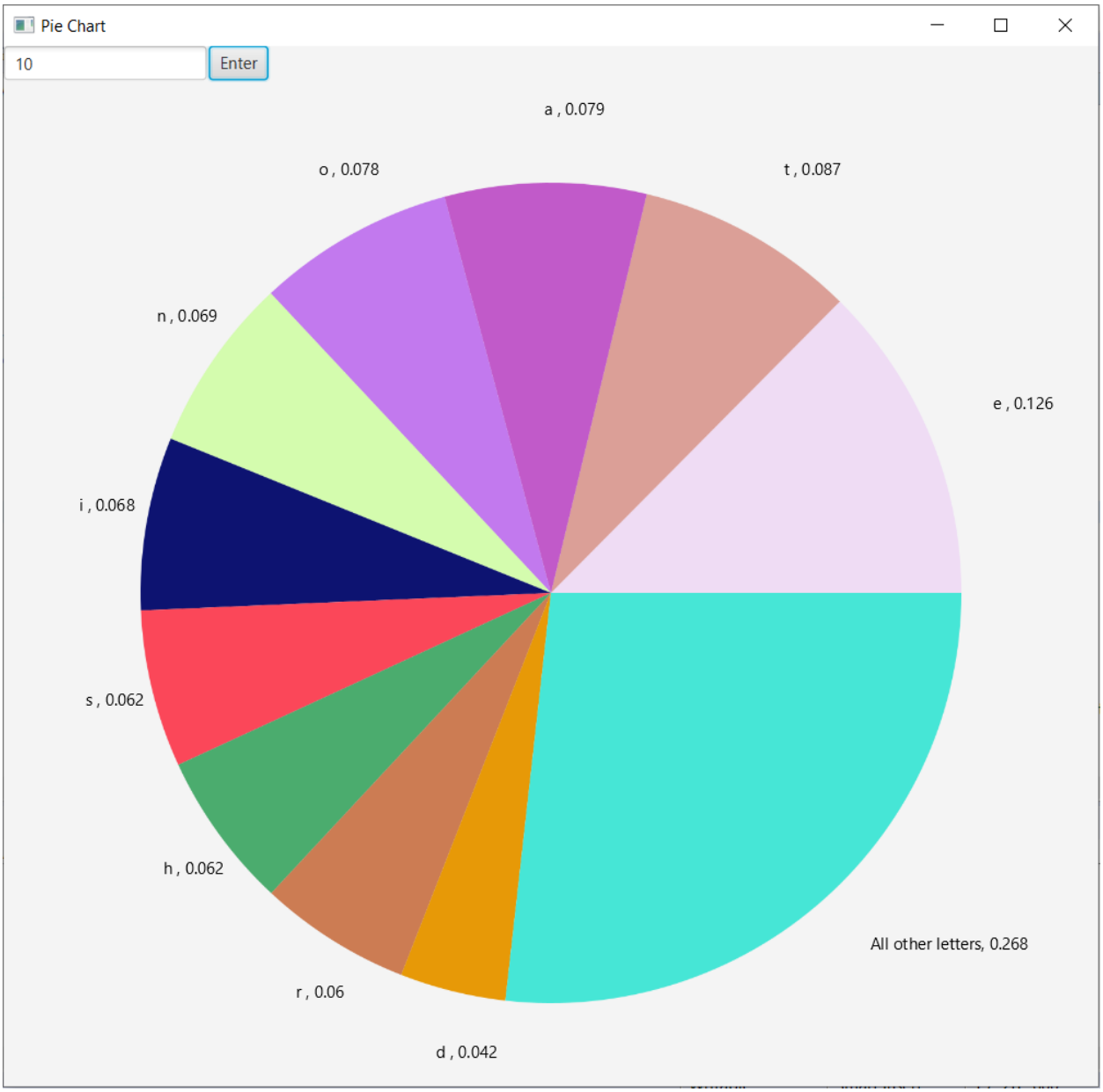
DrawDesign.java

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;


   public class DrawDesign extends Application {

      @Override
      public void start(Stage stage) throws Exception{

          stage.setTitle("Pie Chart");
          Canvas canvas = new Canvas(800, 800);
          GraphicsContext gc = canvas.getGraphicsContext2D();

          String s = "C:\\Users\\good_\\Desktop\\emma.txt";
          HistgramLetters emma = new HistgramLetters(s);
          emma.setLetters_probabilities(emma.letters_frequencies);
          emma.setLetters_angles(emma.letters_probabilities);
          emma.setNSorted(emma.letters_frequencies);

          PieChart chart = new PieChart(emma.getLetter_probabilites(),
emma.getLetters_angles(), emma.getNSorted());

          Pane root = new Pane();
          root.getChildren().add(canvas);
          chart.nInput(gc, root, gc.getCanvas().getWidth(),
gc.getCanvas().getWidth());//Input and button calls drawChart

          Scene scene = new Scene(root);
```

```
        stage.setScene(scene);
        stage.show();
    }


    public static void main(String[] args) {
        launch(args);
    }

}
```

Results:




Pie Chart
10
Enter
a , 0.079
o , 0.078
t , 0.087
n , 0.069
e , 0.126
i , 0.068
s , 0.062
h , 0.062
All other letters, 0.268
r , 0.06
d , 0.042