Data Visualization

——A program that reads a file and draws a Sankey diagram

Rui.Sang 2251576

CONTENTS

OOP design——Encapsulation&Inheritance&Polymorphis&Abstraction

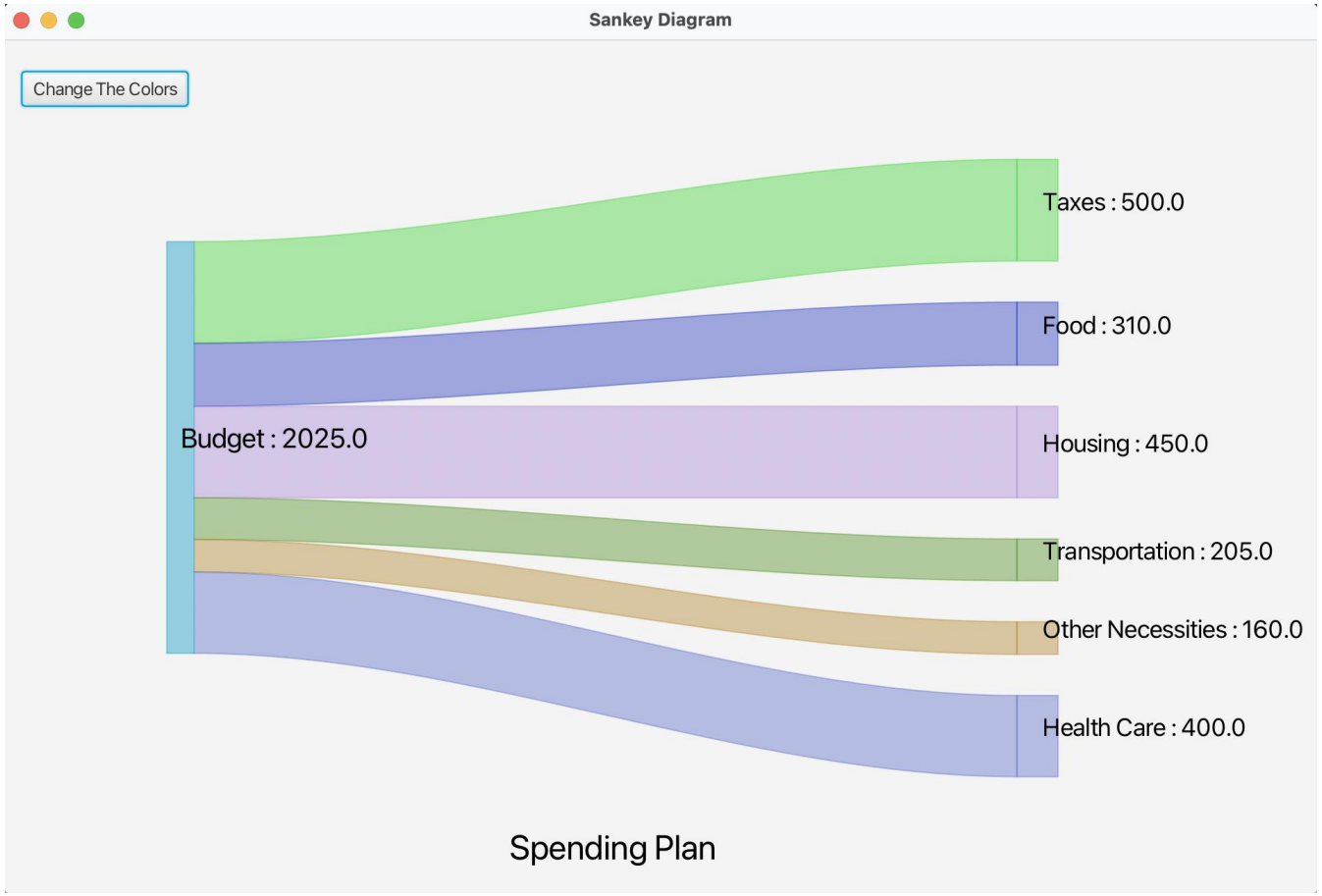
```java
public class FileToSankeyDiagram extends Application {
    @Override
    public void start(Stage primaryStage) {...}

    public static void main(String[] args) {launch(args);}
class FileReader {
    2 usages
    private String title, label;
    3 usages
    private Map<String, Double> dataMap;
    1 usage
    public FileReader(String pathname) {GetDataFromFile(pathname);}
    1 usage
    public String getTitle() { return title; }
    1 usage
    public String getLabel() { return label; }
    1 usage
    public Map<String, Double> getDataMap() { return dataMap; }
    1 usage
    private void GetDataFromFile(String pathname) {...}
    1 usage
    private void processData(List<String> linelist) {...}
    3 usages
    private void processValues(
            String[] values, int index, String string, double cost) {...}
    1 usage
    private boolean isaWord(String str) {...}}
```

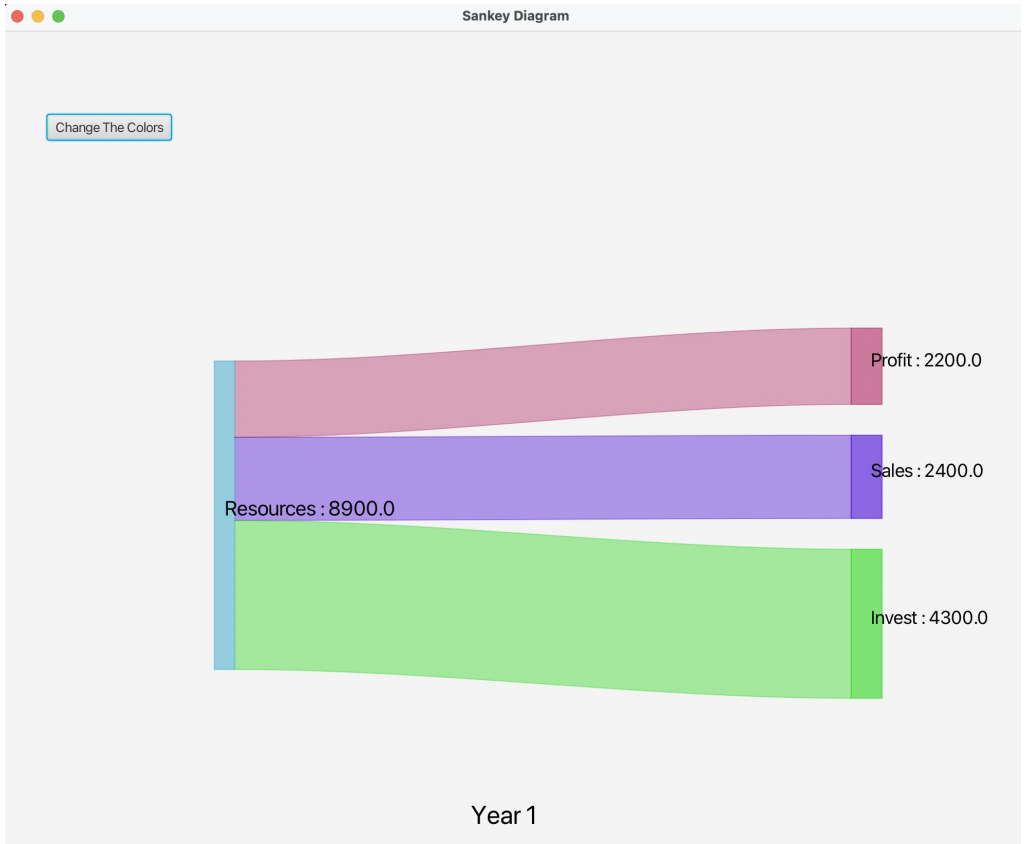
```java
class MyRectangle extends Rectangle {
    1 usage
    private Color strokeColor;
    1 usage
    private Color fillColor;
    1 usage
    public MyRectangle() {}
    2 usages
    public MyRectangle(
            double x, double y,
            double width, double height,
            Color strokeColor, Color fillColor) {...}
}
8 usages
class MyText extends Text {
    1 usage
    private Font font;
    2 usages
    public MyText() {}
    3 usages
    public MyText(double x, double y, String text, Font font) {...}}
```

```java
class SankeyDiagram extends Pane {
    11 usages
    private MyRectangle rectangle;
    3 usages
    private MyText labelTitle,diagramTitle;
    4 usages
    private Group recs,text,curves;
    1 usage
    public SankeyDiagram(String title, String label,
                         Map<String, Double> dataMap) {...}
    1 usage
    public void createSankeyDiagram(String title, String label,
                                    Map<String, Double> dataMap) {...}
    1 usage
    public void changeColors() {...}
    1 usage
    private Color getRandomColor() {...}
    @Override
    public void setWidth(double width) {...}
    @Override
    public void setHeight(double heigth) {...}
    1 usage
    private double calculateTotalSum(Map<String, Double> dataMap) {...}
    1 usage
    private Path createPath(double x, double y,
                            double curveheightchange,
                            MyRectangle r1, double currentHeight) {...}
}
```

* Encapsulation：different classes encapsulate variables and methods—achieve the modularity
* Inheritance：extend the super class and use the constructer in the supper class—promote code reuse
* Polymorphism: override the method in super class
* Abstraction：（detials are in the report）

Diagram Display Algorithm——result display

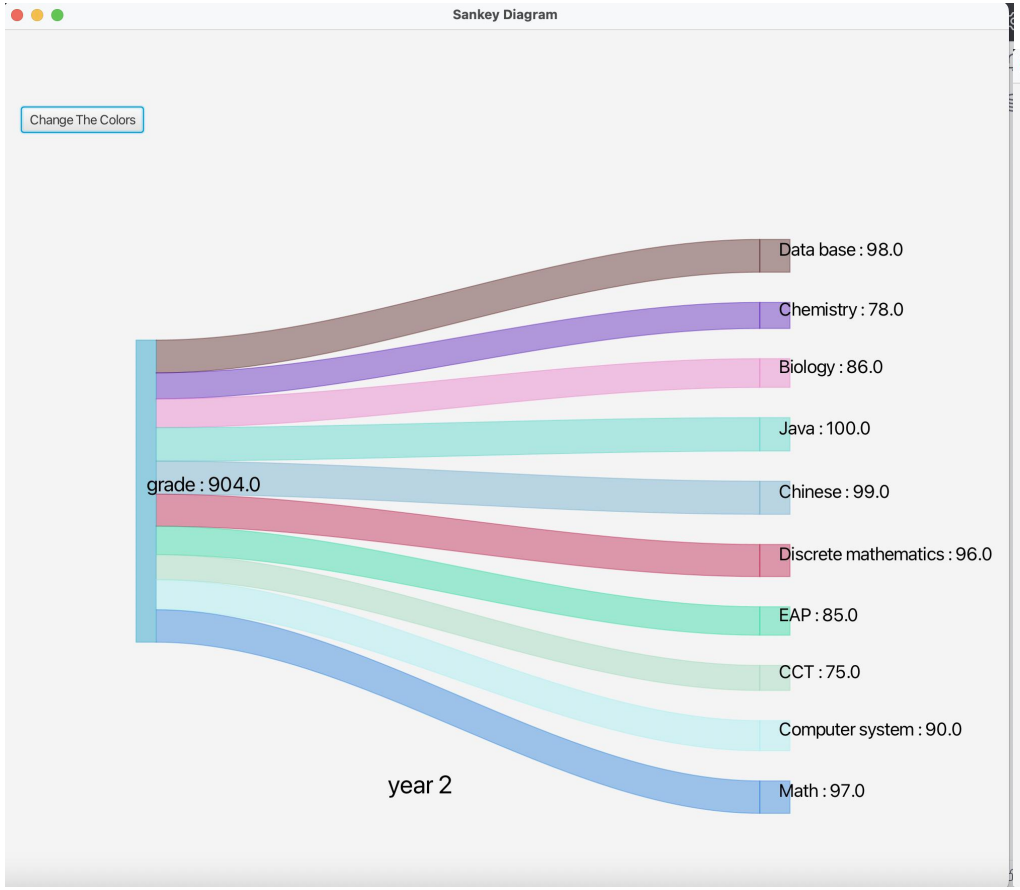


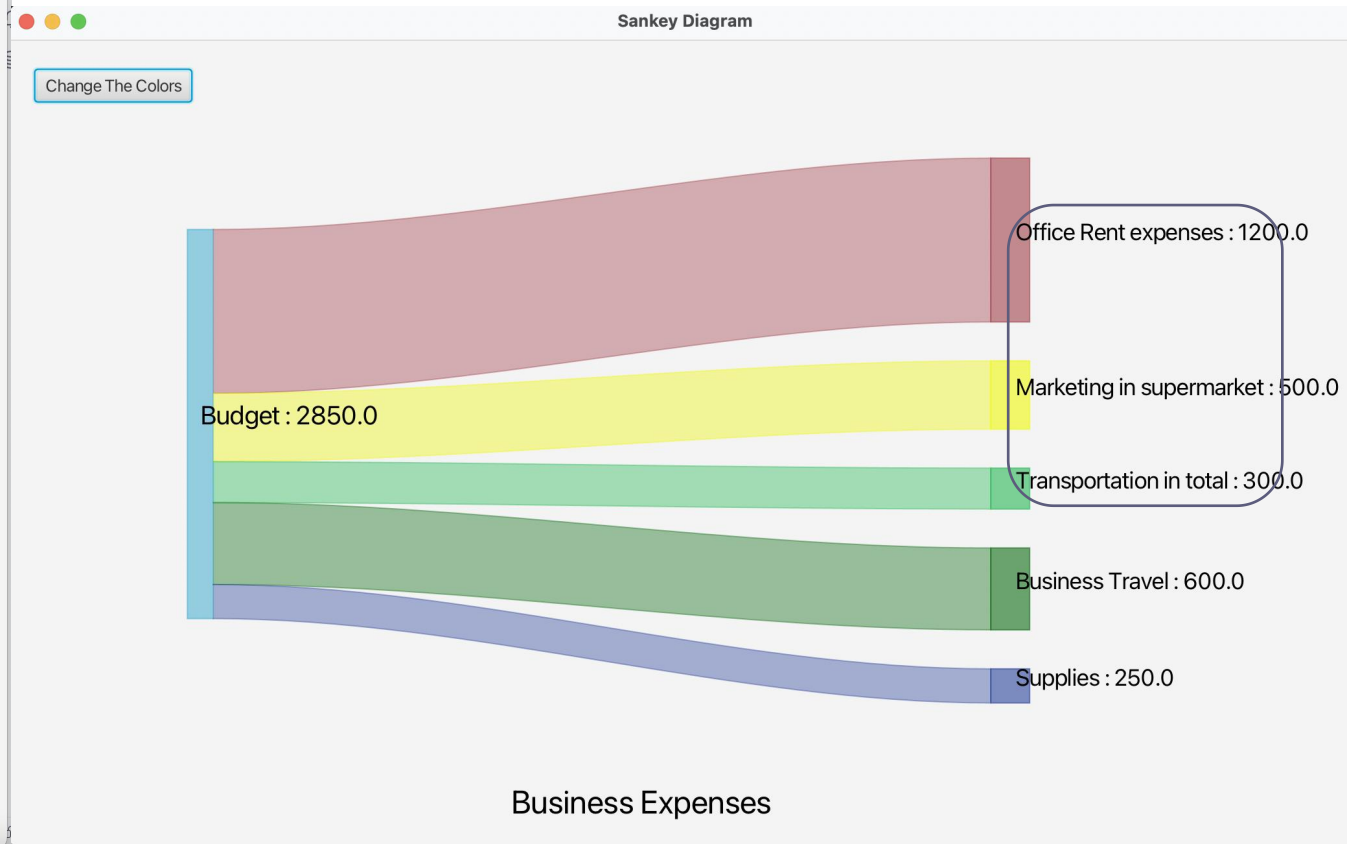


common

short

Diagram Display Algorithm——result display





long

complex string

Diagram Display Algorithm——Data Analysis

* How: HashMap & FOR loop

In the SankeyDiagram class:

```
private double calculateTotalSum(Map<String, Double> dataMap) {
    double sum = 0;
    for (double value : dataMap.values()) {
        sum += value;}
    return sum;}
```

Calculate the sum of the data in the file

In the createSankeyDiagram method (which is In the SankeyDiagram class):

```
double totalHeight = rectangle.getHeight();
double sum = calculateTotalSum(dataMap);

for (String key : dataMap.keySet()) {
    double currentheight = (dataMap.get(key) / sum) * totalHeight;
```

The FOR loop is used to accurately pass the data stored in the map to each small rectangle

Diagram Display Algorithm——Sankey diagram drawing

```java
class SankeyDiagram extends Pane {
    11 usages
    private MyRectangle rectangle;
    3 usages
    private MyText labelTitle, diagramTitle;
    4 usages
    private Group recs, text, curves;
```

……

```java
public void createSankeyDiagram(String title, String label, Map<String, Double> dataMap) {
    diagramTitle = new MyText( x: 350,  y: 550, title,
            Font.font( s: "Courier", FontWeight.BOLD, FontPosture.ITALIC,  v: 25));
    rectangle = new MyRectangle(
            x: 100,  y: 100,  width: 20,  height: 300,
            Color.rgb( i: 51,  i1: 166,  i2: 204,  v: 0.5),
            Color.rgb( i: 51,  i1: 166,  i2: 204,  v: 0.5)
    );
```

……

```java
double totalHeight = rectangle.getHeight();
double sum = calculateTotalSum(dataMap);
label = label + " : " + sum;
labelTitle = new MyText(
        x: rectangle.getX() + rectangle.getWidth() / 2,
        y: rectangle.getY() + rectangle.getHeight() / 2,
        label,
        Font.font( s: "Courier", FontWeight.BOLD, FontPosture.REGULAR,  v: 20)
);
```

……

Diagram Display Algorithm——Sankey diagram drawing

```
class SankeyDiagram extends Pane {
    11 usages
    private MyRectangle rectangle;
    3 usages
    private MyText labelTitle, diagramTitle;
    4 usages
    private Group recs, text, curves;
```

.....

```
for (String key : dataMap.keySet()) {
    double currentheight = (dataMap.get(key) / sum) * totalHeight;

    int randomR = (int) (Math.random() * 256);
    int randomG = (int) (Math.random() * 256);
    int randomB = (int) (Math.random() * 256);

    MyRectangle r1 = new MyRectangle(
            x: X + 600,  y: Y - 60 + recheheightchange,
            width: 30, currentheight,
            Color.rgb(randomR, randomG, randomB,  v: 0.6),
            Color.rgb(randomR, randomG, randomB,  v: 0.6));

    recs.getChildren().add(r1);
```

.....

Call the MyRectangle object to make each small rectangle and add it to the group

Diagram Display Algorithm——Sankey diagram drawing

The text position is determined by the X and Y values of each small rectangle

```
class SankeyDiagram extends Pane {
    11 usages
    private MyRectangle rectangle;
    3 usages
    private MyText labelTitle, diagramTitle;
    4 usages
    private Group recs, text, curves;
```

.....

```
MyText type = new MyText(
        x: r1.getX() + r1.getWidth() / 2,
        y: r1.getY() + r1.getHeight() / 2,
        text: key + " : " + dataMap.get(key),
        Font.font( s: "Courier", FontWeight.BOLD, FontPosture.REGULAR, v: 18));
text.getChildren().add(type);
```

Call the MyText object to make each corresponding text and add it to the group

Diagram Display Algorithm——Sankey diagram drawing

```java
Path path = createPath(X, Y, curveheightchange, r1, currentheight);
Color pathColor = Color.rgb(randomR, randomG, randomB, v: 0.4);
path.setStroke(pathColor);
path.setFill(pathColor);

curves.getChildren().add(path);
```

```java
class SankeyDiagram extends Pane {
    11 usages
    private MyRectangle rectangle;
    3 usages
    private MyText labelTitle, diagramTitle;
    4 usages
    private Group recs, text, curves;

    ......
```

Call the Path object to make each corresponding curve and add it to the group

```java
private Path createPath(double x, double y,
                        double curveheightchange,
                        MyRectangle r1, double currentHeight) {
    //上曲线的起始点和控制点
    MoveTo moveTo1 = new MoveTo(x, v1: y + curveheightchange);
    double endX = r1.getX();
    double endY = r1.getY();
    //Determine the amount of change at the control point确定控制点的变化量
    double changeX = Math.abs(x - endX);
    double controlX1 = moveTo1.getX() + changeX / 3;
    double controlY1 = moveTo1.getY();
    double controlX2 = endX - changeX / 3;
    double controlY2 = endY;

    CubicCurveTo curveTo1 = new CubicCurveTo(
            controlX1, controlY1,
            controlX2, controlY2,
            endX, endY
    );
```

```java
    //下曲线的起始点和控制点
    LineTo lineTo1 = new LineTo(r1.getX(), v1: r1.getY() + r1.getHeight());
    double startX = x;
    double startY = y + curveheightchange + currentHeight;
    double controlX3 = startX + changeX / 3;
    double controlY3 = startY;
    double controlX4 = lineTo1.getX() - changeX / 3;
    double controlY4 = lineTo1.getY();

    CubicCurveTo curveTo2 = new CubicCurveTo(
            controlX4, controlY4,
            controlX3, controlY3,
            startX, startY
    );

    Path path = new Path();
    path.getElements().addAll(moveTo1, curveTo1, lineTo1, curveTo2);
    return path;
}
```

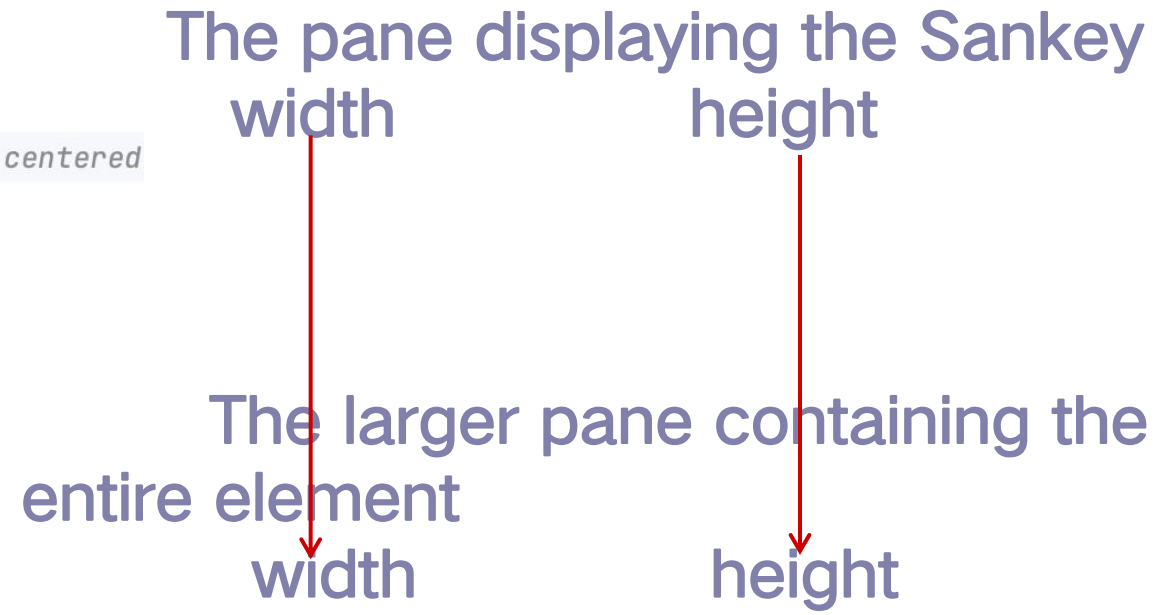
The creatPath method creates a path through the upper and lower curves and four control points.

Display while Resizing Algorithm

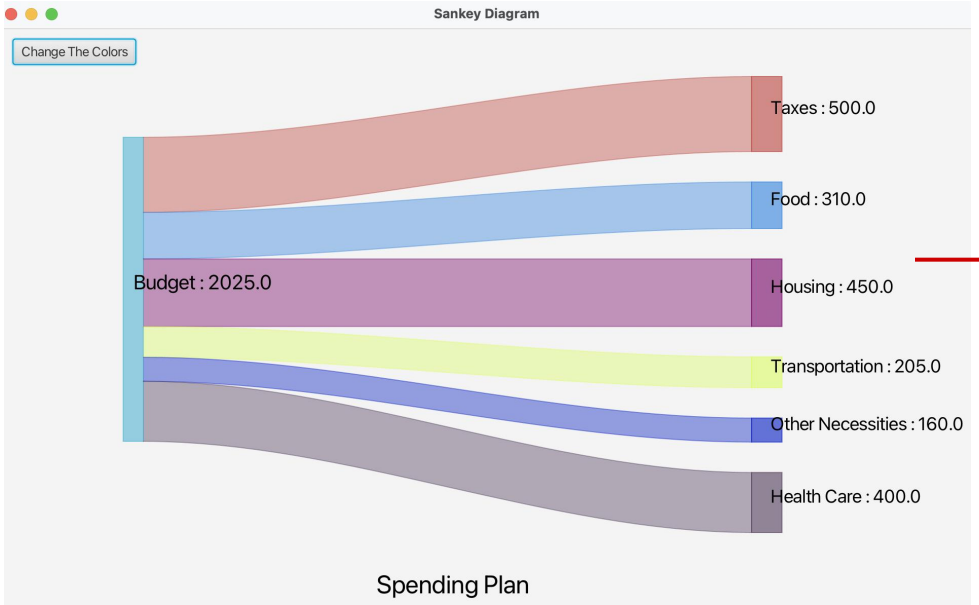
- **Main Method：DoubleBinding**

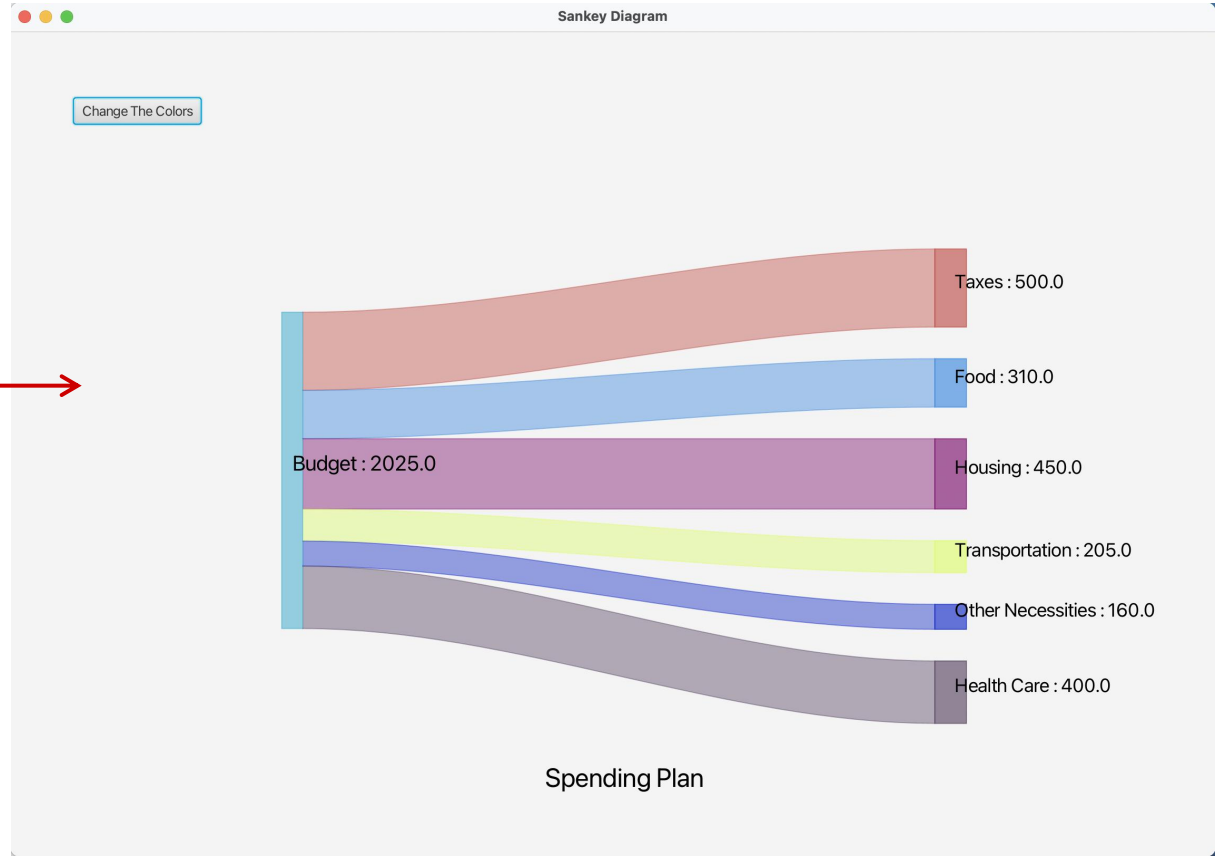
```
// 使用绑定属性使图形保持居中Use binding properties to keep the graph centered
pane.translateXProperty().bind(Bindings.createDoubleBinding(
        () -> (root.getWidth() - pane.getWidth()) / 2,
        root.widthProperty(), pane.widthProperty()
));
pane.translateYProperty().bind(Bindings.createDoubleBinding(
        () -> (root.getHeight() - pane.getHeight()) / 2,
        root.heightProperty(), pane.heightProperty()
));

Scene scene = new Scene(root, v: 1000, v1: 1000);
```

……



Display while Resizing Algorithm





Additional Features—The Color change button

```java
// 添加按钮Add button
Button colorButton = new Button( s: "Change The Colors");
colorButton.setLayoutX(10);
colorButton.setLayoutY(10);
colorButton.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        // 调用changeColors方法更换颜色 Call the changeColors
        pane.changeColors();
    }
});
Group newpane = new Group();
newpane.getChildren().addAll(pane, colorButton);
```

.....

```java
public void changeColors() {
    List<MyRectangle> rectangles = new ArrayList<>();
    List<Path> paths = new ArrayList<>();
    for (Node node : recs.getChildren()) {
        if (node instanceof MyRectangle) {
            rectangles.add((MyRectangle) node);}}
    for (Node node : curves.getChildren()) {
        if (node instanceof Path) {
            paths.add((Path) node);}}
    for (int i = 0; i < rectangles.size(); i++) {
        MyRectangle rectangle = rectangles.get(i);
        Path path = paths.get(i);
        Color newColor = getRandomColor();
        rectangle.setStroke(newColor);
        rectangle.setFill(newColor);
        path.setStroke(newColor);
        path.setFill(newColor);}}
```

```java
private Color getRandomColor() {
    int randomR = (int) (Math.random() * 256);
    int randomG = (int) (Math.random() * 256);
    int randomB = (int) (Math.random() * 256);
    return Color.rgb(randomR, randomG, randomB, v: 0.4);
}
```

.....

* Add buttons and set properties
* Override and call the method that changes the color
* Adds to a new pane and joins to the scene

* Use two for loops to add both the rectangle and path to the new list
* Use a for loop so that it is added with the newly generated color accordingly

* A private method that generates random colors

Additional Features—The Color change button

File Handling

```java
class FileReader {
    2 usages
    private String title, label;
    3 usages
    private Map<String, Double> dataMap;
    1 usage
    public FileReader(String pathname) {GetDataFromFile(pathname);}
    1 usage
    public String getTitle() { return title; }
    1 usage
    public String getLabel() { return label; }
    1 usage
    public Map<String, Double> getDataMap() { return dataMap; }
    1 usage
    private void GetDataFromFile(String pathname) {...}
    1 usage
    private void processData(List<String> linelist) {...}
    3 usages
    private void processValues(
            String[] values, int index, String string, double cost) {...}
    1 usage
    private boolean isaWord(String str) {...}
```

* File Reading:
Put each line into a list

```java
private void processData(List<String> linelist) {
    dataMap = new HashMap<>();
    //处理文件中的相关信息
    for (int i = 2; i < linelist.size(); i++) {
        String line = linelist.get(i);
        String[] values = line.split( regex: " ");
        processValues(values,  index: 0,  string: "",  cost: 0);
    }
}
3 usages
private void processValues(
        String[] values, int index, String string ,double cost) {
    if (index >= values.length) {
        dataMap.put(string, cost);
        return;
    }
    String currentValue = values[index];
    if (isaWord(currentValue)) {
       string += " " + currentValue;
        processValues(values,  index: index + 1, string, cost);
    } else {
        cost = Double.parseDouble(currentValue);
        processValues(values,  index: index + 1, string, cost);
    }
}
1 usage
private boolean isaWord(String str) {
    for (char c : str.toCharArray()) {
        if (!Character.isLetter(c)) {
            return false;
        }
    }
    return true;
}}
```

* Data parsing and Verification

Read the file add every lines to a list

↓

Split every lines to a new array

↓

In every array use a method to determine whether an element is a word or a number

↓

then use recursion to get the name of each class and its corresponding data very accurately

Exception Handling

```
    try {
        Scanner input = new Scanner(file);
        while (input.hasNextLine()) {
            String line = input.nextLine();
            linelist.add(line);
        }
        this.title = linelist.get(0);
        this.label = linelist.get(1);
        processData(linelist);
    } catch (IOException ioe) {
        System.out.println(ioe.getMessage());
    }
}
```

.....

Type——IOException:
* May occur during a file input/output operation.
* In the code, when reading a file through Scanner, we use the Scanner constructor and the nextLine() method, whose calls may throw IOException

Handling Policy:
* Using a try-catch block
* First, it tries to run in the try block. If the code in the try block raises IOException, it jumps to the catch block. The exception message for the caught IOException is printed with System.out.println(ioe.getMessage())

Exception Handling

```java
private void GetDataFromFile() {
    File file = new File( pathname: " ");
    List<String> linelist = new ArrayList<>();
    //先读取文件，将文件内容按line 分进list，并且判断是否在读取的时候有异常
    try {
        Scanner input = new Scanner(file);
        while (input.hasNextLine()) {
            String line = input.nextLine();
            linelist.add(line);
        }
        this.title = linelist.get(0);
        this.label = linelist.get(1);
        processData(linelist);
    } catch (IOException ioe) {
        System.out.println(ioe.getMessage());
    }
}
```

```
(No such file or directory)
Exception in Application start method
Exception in thread "main" java.lang.RuntimeException Create breakpoint : Exception in
 Application start method
 ......
```

The end

Thank you and Happy New Year!!