Bilkent University

Department of Computer Engineering

CS 315 Programming Languages

Design a Drawing Figure Language

Part I Report

Project Group Member Name:
Emir Acımış, 21201233
Izel Gürbüz, 21301018
Mustafa Culban, 21301187

Supervisor: Ramazan Gökberk Cinbiş
Language Name: Drawer++

Nov 28, 2016

Introduction

Drawer++ is a drawing figure language. In this language the user can draw some special figures like rectangle, oval, line and composite shapes. The syntax of the language is inspired from the C++ language's syntax. The language support create some shapes, whit this shapes they can customize these shapes like filling with color or changing the size of them. The properties consist of name and values.

Drawer++ also support the simple known primitive types like string, integer, float and boolean. The language also support the simple calculation like summation, multiplication, division and subtraction.

With the help of the language Drawer++ users can create composite shapes also which means they can also extend the shapes of the program. For example a rectangle can be change to different shapes or in a rectangle, different shapes can be added to inside or outside.

The properties and the syntax of the Drawer++ is define as follow;

1) **Defining Shape**

   A) **Defining Rectangle**

      a) **Default Rectangle Constructor**

      In Default constructer the programmer do not need any property to define inside the rectangle. Whit this way the user create a default rectangle that system define. For Example;

      begin

            Shape s1

            s1 = Rectangle()

      end

When the programmer use this constructer, s/he can use the functions that defining for the Shape object. The system syntax is working like this;

```
begin
        Shape s1
        s1 = Rectangle()
        s1.color = RGB(0,0,225,0.8)
        s1.location.xValue = 10
        s1.location.yValue = 40
        s1.height = 10
        s1.width = 20
        s1.StrokeWidth = 10
        s1.rounded = true
end
```

b) Rectangle Constructor

In normal constructor the programmer can define the specific values for the rectangle. For Example;

```
begin
        Location l1 =  Location (10,40)
        integer w1 = 10
        integer h1 = 20 // height
        Set RGB = Set {225,225,225,0.8};
        Color c1 =  Color ( RGB) // color
        integer sw1 = 10 // stroke width
        boolean r1 = false //rounded
        Set a1 =  {l1.xValue,l1.yValue,w1,h1}
```

```
Shape s2 = Rectangle(c1, a1, sw1, r1)
```

end

B) Defining Oval

a) Default Oval Constructor

When we create oval (it is same as create a rectangle) the syntax below is create a default shape oval.

```
begin
    Shape s3
    s3 = Oval()
end
```

When programmer change the constructer the functions will be implemented in this way;

```
begin
    Shape s3
    s3 = Oval()
    s3.color = RGB(0,0,225,0.8)
    s3.location.xValue = 10
    s3.location.yValue = 40
    s3.size = (20,30)
    s3.StrokeWidth = 10
end
```

b) Oval Constructor

Without default constructor the Oval can be implemented like rectangle but there are some difference from the rectangle, for example an oval shape cannot be rounded. For Example;

```
begin
        Location l1 =  Location (10,40)
        integer w1 = 10
        integer h1 = 20 // height
        Set RGB = Set {225,225,225,0.8};
        Color c1 =  Color ( RGB) // color
        integer sw1 = 10 // stroke width
        boolean r1 = false //rounded
        Set a1 =  {l1.xValue,l1.yValue,w1,h1}

        Shape s4 =  Oval(c1, a1, sw2)
end
```

C) Defining Line

a) Default Line Constructor

Line also define same as the rectangle and oval for the default constructor and with below syntax the system create a simple line. For Example;

```
begin
        Shape s5
        s5 = Line()
end
```

With the help of the functions for the Line the default line can be change with below syntax. For Example;

```
begin
        Shape s5
        s5 = Line()
        s5.color = RGB(0,0,225,0.8)
        s5.location.xValue = 10
        s5.location.yValue = 40
        s5.direction = "N"
        s5.height = 20
        s5.strokeWidth = 2
        s5.arrow = true // if it is true it is the start arrow
        s5.arrowSize = 4 // giving the arrow size
end
```

b) Line Constructor

Without default constructor the Line can be implemented like rectangle and oval. But there are some difference for the line like the line can have an arrow or the Line does not have width. For Example;

```
begin
        Location l1 = Location (10,40)
        Direction d1 = Direction ("N")
        integer h1 = 20 // height
        Set a1 =  {l1.xValue,l1.yValue,h1,d1} // set of attributes
        Set RGB = Set {225,225,225,0.8};
        Color c1 =  Color ( RGB) // color
        integer sw1 = 2 //stroke
        boolean startArrow = true // is start arrow or end arrow
        int arrowSize = 4 // arrow head size
        Shape s6 = Line(c1,a1,sw3,startArrow,arrowSize)
end
```

D) Defining Composite Shape

a) Default Composite Shape Constructor

Composite Shape is also shape but composite shape not a proper shape like rectangle and oval. The composite shape is a mixture of the shape and the system give a default shape with this syntax;

begin

Shape s7 = CompositeShape();

end

Composite shape can be change with different properties like we describe in the rectangle, oval and line. For Example;

b) Composite Shape Constructor

Composite shape is also define without default constructor which means first define the properties of shape and then create the shape;

2) Defining Functions

A) Draw function

Draw function is the most important function for the language because the function is draw the function to the screen and the user see the shape that they create.

For Example; (The example of this function variables thinking that already initialize)

begin

.

.

.

draw.s1 // draw the shape that we already initialize

draw.s2

draw.s3

draw.s4

draw.s5

.

.

.

end

B) Fill Color Function

This function allow the user for filling the color inside the shape that user draw.

For Example; (The example of this function variables thinking that already initialize)

begin

.

.

.

s1.fillColor(RGB) // It is time to fill inside

s2.fillColor(RGB) // It is time to fill inside

s3.fillColor(RGB) // It is time to fill inside

.

.

.

end

C) Fil State Function