Computer Graphics Programming

SID: 1506711

May 2, 2019

1 Extensions to GrafPak

1.1 Additional shapes and shape creation

To extend shape creation I have added Circles and Triangles which can both be created in the window by selecting them from the create menu and then clicking in two different places for a circle, or 3 places for a triangle. Rubber band drawing has also been implemented for drawing squares which works by left clicking where you want the first point of the square to be and moving the mouse to adjust the size while keeping the left mouse button held down and releasing once you have the square at the size and orientation you want. Squares can still be drawn by selecting two points as long as the mouse isn't moved when releasing the left mouse button on the first point and rubber banding will not work when selecting the second point. I attempted to add in rubber band drawing for circles however I couldn't get that to work with redrawing the circle while moving the mouse.

Circles are drawn using the Bresenham Circle Drawing Algorithm based on two initial points which determine the size of the radius. The first point is the center point and the second point is a point along the outer edge of the circle. Once these points are determined, the algorithm then calculates new points around the circle at 45°intervals and repeated until the full circle is made.

Triangles are drawn by determining the 3 corners of the triangle and then drawing lines between the 3 points to form the walls.

one problem that exists is that when drawing a square with rubber banding it erases parts of any previously drawn shapes that it overlaps while the square is being adjusted and redrawn.

1.2 Selection of shapes

Shapes are selected by using the left and right arrow keys to loop through the shapes that are drawn in the window. I couldn't work out how to select a specific shape using the mouse to click on an edge of a shape so that isn't implemented in the program. The rotation of selection is achieved by looping through a list of shapes that is defined in the GrafPak class. Every shape that gets created is added to this list and gets removed when deleted. The shape that is currently selected is represented by it's lines being blue instead of black. To deselect the selected shape, the esc key can be pressed.

```
private void selectShape(int direction)
{
    if (shapes.Count < 1)
    {
        return;
    }

    if (selectedShape != null)
    {
        redraw(selectedShape, Brushes.Black);

        int curIndex = shapes.IndexOf(selectedShape);

        if (direction > 0)
        {
            // right (+1)
            // curIndex + 1
            if (curIndex + direction <= shapes.Count -1)
            {
                selectedShape = shapes[curIndex + direction];
            }
            else
            {
                selectedShape = shapes[0];
            }
        }
```

```
        else
        {
            // left (-1)
            // curIndex + (-1)
            if (curIndex + direction >= 0)
            {
                selectedShape = shapes[curIndex + direction];
            }
            else
            {
                selectedShape = shapes[shapes.Count - 1];
            }
        }
    }
    else
    {
        selectedShape = shapes[0];
    }

    redraw(selectedShape, Brushes.Blue);
    this.transformItem.Enabled = true;
}
```

This is the method that is used to select shapes, if the direction integer is 0 then the selection will go to the left of the currently selected shape, otherwise it will go to the right. if no shape is selected, the first shape in the shapes list will be selected

1.3 Deleting shapes

Shapes can be deleted by selecting them and pressing the delete key on the keyboard. This is done by redrawing over the shape with white lines to match the background. The shape is also removed from the list of shapes to avoid the program from selecting it once it has been deleted by the user.

```
private void keyUp(object sender, KeyEventArgs e)
{
    switch(e.KeyCode)
    {
        // select previous shape
        case Keys.Left:
            selectShape(-1);
            break;
        // select next shape
        case Keys.Right:
            selectShape(1);
            break;
        // delete selected shape
        case Keys.Delete:
            if (selectedShape != null)
            {
                redraw(selectedShape, Brushes.White);
                shapes.Remove(selectedShape);
                selectedShape = null;
                this.transformItem.Enabled = false;
            }
            break;
        // deselect shape
        case Keys.Escape:
            redraw(selectedShape, Brushes.Black);
            selectedShape = null;
            this.transformItem.Enabled = false;
            break;
    }
}
```

This method is used as an event listener for when the user presses a key on their keyboard. It is called once the key has been released. Depending on which key is pressed, a different action will be performed. For left and right arrow keys, it will rotate through the list of shapes for selection, for the delete key if will delete the shape that is currently selected and for the esc key it will deselect the currently selected shape.