# Assignment 8 Project

## Final Project Type IV
## (Computer Graphics and Animation)

Peter Jukrat

Sanjana Lawande(15.18)
Neil Kadam(15.19)
Sangram Vasantrao Thorat(15.32)

# Project Summary

Problem Statement: Use one or more of the exercises in Chapters 14 and 15 to explore computer graphics and animation. The project was divided into three parts, assigning each member with a question from the chapters 14 & 15.

The three questions are as below.

**15.18 (Sanjana Lawande)** - (Move a rectangle using mouse) Write a program that displays a rectangle. You can point the mouse inside the rectangle and drag(i.e., move the mouse pressed) the rectangle wherever the mouse goes. The mouse point becomes the center of the rectangle.

- I have used the following functions on Rectangle class -
  setX() to set the x coordinate of the center of the rectangle
  setY() to set the y coordinate of the center of the rectangle
  setFill() to set the color of the rectangle as YELLOW
  setStroke() to set the border of the rectangle as RED

**Code for 15.18-**

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.Pane;
import javafx.scene.layout.StackPane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.stage.Stage;

public class Project_C15E18 extends Application {
   @Override
   public void start(Stage primaryStage) {
      double width = 500;
      double height = 400;

      Rectangle rec = new Rectangle(100, 80, 100, 80);
      rec.setFill(Color.YELLOW);
      rec.setStroke(Color.RED);

      Pane pane = new Pane(rec);
```

```
        rec.setOnMouseDragged(e-> {
            rec.setX(e.getX() - rec.getWidth() / 2);
            rec.setY(e.getY() - rec.getHeight() / 2);
        });

        primaryStage.setScene(new Scene(pane, width, height));
        primaryStage.setTitle("MOVE THE RECTANGLE USING MOUSE");
        primaryStage.show();
    }

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

**15.19 (Neil Kadam)** - (Game: eye-hand coordination) Write a program that displays a circle of radius 10 pixels filled with a random color at a random location on a pane. When you click the circle, it disappears and a new random-color circle is displayed at another random location. After 20 circles are clicked, display the time spent in the pane.

→ This application can be run like any JAVAFX program. I have divided the game into three panes. The START page, HELP page, and the GAME page.

- Start Page: This includes three buttons: Start, How to Play, and Exit
  The Start button starts the game and the timer. How to Play takes you to the help pane where you can see the instructions about how to play the game. And the exit button quits the application.The Exit button quits the application.

- Help Page: Clicking on How to Play takes you to this pane which contains instructions about how to play the game and includes a BACK button to go back to the main menu.

- Game Page: Once you press the start button the program will take you to the main game, which when finished will show the results and a back button.

**Code for 15.19** -

```
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
```

```
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class C15E19CircleGame extends Application{

	public void start(Stage primaryStage){
		BorderPane root = new BorderPane();
		counter = 1;

		Group g1 = new Group();
		Scene sc1 = new Scene(root, 500, 500);
		root.setCenter(g1);

		Label menu = new Label("Aim Trainer");
		Button start = new Button("Start");
		Button help = new Button("How to Play");
		Button exit = new Button("Exit");
		Label gameScr = new Label("Game Screen");

		menu.setTranslateY(15);
		gameScr.setTranslateY(15);
		start.setTranslateY(50);
		help.setTranslateY(80);
		exit.setTranslateY(110);

		g1.getChildren().addAll(menu, start, help, exit);

		start.setOnMouseClicked(e -> { game(primaryStage); });
		help.setOnMouseClicked(e -> {help(primaryStage);});
		exit.setOnMouseClicked(e -> {primaryStage.close();});

		primaryStage.setScene(sc1);
		primaryStage.setTitle("Circle Game");
	primaryStage.show();
	}

	public void help(Stage primaryStage){
		BorderPane root = new BorderPane();

		Group g1 = new Group();
		Scene sc1 = new Scene(root, 500, 500);
		root.setCenter(g1);

		Circle circle = new Circle(200, 50, 10);
		//Filling the circle with random color using RGB values
		circle.setFill(Color.color(Math.random(), Math.random(), Math.random(), 1.0));
		Label desc = new Label("Once the game starts click on the circles as they appear"
				+ "\nThe circles look like this ->");
		Button back = new Button("Back");
```

```
                Label gameScr = new Label("Game Screen");

                desc.setTranslateY(15);
                gameScr.setTranslateY(15);
                back.setTranslateY(80);

                g1.getChildren().addAll(desc, back, circle);

                back.setOnMouseClicked(e -> { start(primaryStage); });

                primaryStage.setScene(sc1);
                primaryStage.setTitle("Circle Game");
        primaryStage.show();
        }

        static int counter = 1;

        public void game(Stage primaryStage) {
                //setting the game window size
                final double width = 500.0;
                final double height = 500.0;

                //initiating timer for calculating the final time
                long start = System.currentTimeMillis();
                Pane pane = new Pane();

                //Creating the circle at random position using math.random()
                Circle circle = new Circle(Math.random() * 400, Math.random() * 400, 10);
                //Filling the circle with random color using RGB values
                circle.setFill(Color.color(Math.random(), Math.random(), Math.random(), 1.0));

                //clearing the pane and popping and new circle if counter is not reached.
                //else showing the time spent in msec.
                circle.setOnMouseClicked(e -> {
                        //can reduce the number of click here:
                        if (counter == 20) {
                                pane.getChildren().clear();
                                long end = System.currentTimeMillis();
                                long total = end - start;
                                Button back = new Button("Back to Main Menu");
                                back.setTranslateY(180);
                                back.setTranslateX(180);
                                back.setOnMouseClicked(b -> { start(primaryStage); });
                                Text time = new Text(width/2 - 120, height/2, String.format("Well Done!
Time spent is %d milliseconds", total));
                                pane.getChildren().addAll(time, back);
                        }
                        else {
                                circle.setCenterX(Math.random() * 400 + 50);
```

```
                                circle.setCenterY(Math.random() * 400 + 50);
                                circle.setFill(Color.color(Math.random(), Math.random(), Math.random(),
1.0));
                                counter++;
                        }
                }
                                );

                //setting up the scene
                pane.getChildren().add(circle);
                Scene scene = new Scene(pane, width, height);
                primaryStage.setTitle("Circle Game");
                primaryStage.setScene(scene);
                primaryStage.show();
        }
        public static void main(String[] args) {
                launch(args);
        }
}
```

**15.32 (Sangram Vasantrao Thorat)** - (Control a clock) - Write a program that lets the user control the clock with the Start and Stop buttons. This program will display the Analog clock that shows the current time with hour, minute and second clock hand. Also the program lets users control the clock with the Start and Stop buttons.

→ Project Structure is divided into 2 parts - ClockPane class and Program Test class. Instructions.

This program will have Start and Stop buttons. When the user runs the program, the live clock is displayed on screen with Start and Stop buttons. When the user clicks on the Stop button, clock is stopped for that time and again when click on Start button, clock will resume from current live time.

This project has 2 java files:
Code for 15.32:

1. **ClockPane.java**

```
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.collections.ObservableList;
import javafx.geometry.Point2D;
import javafx.scene.Node;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.shape.Line;
```

```
import javafx.scene.text.Text;
import javafx.util.Duration;

import java.util.Calendar;
import java.util.Collections;
import java.util.GregorianCalendar;

public class ClockPane extends Pane {

	private int hour;
	private int minute;
	private int second;
	private boolean hourHandVisible = true;
	private boolean minuteHandVisible = true;
	private boolean secondHandVisible = true;
	private Timeline timeline;

	// Clock pane's width and height
	private double w = 500, h = 500;

	/** Construct a default clock with the current time */
	public ClockPane() {
		setPrefHeight(h);
		setPrefWidth(w);
		timeline = new Timeline(new KeyFrame(Duration.seconds(1), e -> update()));
		timeline.setCycleCount(Timeline.INDEFINITE);
		setCurrentTime();
	}

	/** Construct a clock with specified hour, minute, and second */
	public ClockPane(int hour, int minute, int second) {
		this();
		this.hour = hour;
		this.minute = minute;
		this.second = second;
		paintClock();
	}

	public ClockPane(int hour, int minute, int second, double width, double height) {
		this(hour, minute, second);
		this.w = width;
		this.h = height;
		paintClock();
	}

	/** Return hour */
	public int getHour() {
		return hour;
	}
```

```
/** Set a new hour */
public void setHour(int hour) {
	this.hour = hour;
	paintClock();
}

/** Return minute */
public int getMinute() {
	return minute;
}

/** Set a new minute */
public void setMinute(int minute) {
	this.minute = minute;
	paintClock();
}

/** Return second */
public int getSecond() {
	return second;
}

/** Set a new second */
public void setSecond(int second) {
	this.second = second;
	paintClock();
}

/** Return clock pane's width */
public double getW() {
	return w;
}

/** Set clock pane's width */
public void setW(double w) {
	this.w = w;
	paintClock();
}

/** Return clock pane's height */
public double getH() {
	return h;
}

/** Set clock pane's height */
public void setH(double h) {
	this.h = h;
	paintClock();
```

```
}

public boolean isHourHandVisible() {
	return hourHandVisible;
}

public void setHourHandVisible(boolean hourHandVisible) {
	this.hourHandVisible = hourHandVisible;
	paintClock();
}

public boolean isMinuteHandVisible() {
	return minuteHandVisible;
}

public void setMinuteHandVisible(boolean minuteHandVisible) {
	this.minuteHandVisible = minuteHandVisible;
	paintClock();
}

public boolean isSecondHandVisible() {
	return secondHandVisible;
}

public void setSecondHandVisible(boolean secondHandVisible) {
	this.secondHandVisible = secondHandVisible;
	paintClock();
}

/* Set the current time for the clock */
public void setCurrentTime() {
	// Construct a calendar for the current date and time
	Calendar calendar = new GregorianCalendar();

	// Set current hour, minute and second
	this.hour = calendar.get(Calendar.HOUR_OF_DAY);
	this.minute = calendar.get(Calendar.MINUTE);
	this.second = calendar.get(Calendar.SECOND);

	paintClock(); // Repaint the clock
}

/** Paint the clock */
private void paintClock() {
	// Initialize clock parameters
	double clockRadius = Math.min(w, h) * 0.8 * 0.5;
	double centerX = w / 2;
	double centerY = h / 2;
	Point2D center = new Point2D(centerX, centerY);
```

```
		// Draw circle
		Circle circle = new Circle(centerX, centerY, clockRadius);
		circle.setFill(Color.WHITE);
		circle.setStroke(Color.BLACK);

		// Draw time numbers
		Text[] texts = new Text[12];
		for (int i = 0; i < 12; i++) {
			int time = (i + 3 > 12) ? i + 3 - 12 : i + 3;
			Point2D b = new Point2D(centerX + clockRadius * Math.cos(i * 2 * Math.PI / 12),
					centerY + clockRadius * Math.sin(i * 2 * Math.PI / 12));
			b = getPointBCloserToA(center, b, 0.82);
			texts[i] = new Text(b.getX() - (clockRadius * 0.03125), b.getY() + (clockRadius *
0.025), "" + time);
		}

		// Draw dashes
		Line[] dashes = new Line[60];
		for (int i = 0; i < dashes.length; i++) {
			Point2D start = new Point2D(centerX + clockRadius * Math.cos(i * 2 * Math.PI /
60),
					centerY + clockRadius * Math.sin(i * 2 * Math.PI / 60));
			double coefficient = (i % 5 == 0) ? 0.91 : 0.955;
			Point2D end = getPointBCloserToA(center, start, coefficient);
			dashes[i] = new Line(start.getX(), start.getY(), end.getX(), end.getY());
		}

		// Draw second hand
		double sLength = clockRadius * 0.8;
		double secondX = centerX + sLength * Math.sin(second * (2 * Math.PI / 60));
		double secondY = centerY - sLength * Math.cos(second * (2 * Math.PI / 60));
		Line sLine = new Line(centerX, centerY, secondX, secondY);
		sLine.setStroke(Color.RED);
		sLine.setVisible(isSecondHandVisible());

		// Draw minute hand
		double mLength = clockRadius * 0.65;
		double xMinute = centerX + mLength * Math.sin(minute * (2 * Math.PI / 60));
		double minuteY = centerY - mLength * Math.cos(minute * (2 * Math.PI / 60));
		Line mLine = new Line(centerX, centerY, xMinute, minuteY);
		mLine.setStroke(Color.BLUE);
		mLine.setVisible(isMinuteHandVisible());

		// Draw hour hand
		double hLength = clockRadius * 0.5;
		double hourX = centerX + hLength * Math.sin((hour % 12 + minute / 60.0) * (2 * Math.PI /
12));
		double hourY = centerY - hLength * Math.cos((hour % 12 + minute / 60.0) * (2 * Math.PI /
12));
```

```
                Line hLine = new Line(centerX, centerY, hourX, hourY);
                hLine.setStroke(Color.GREEN);
                hLine.setVisible(isHourHandVisible());
                // Draw time HH:MM:SS
                String s = "" + getHour() + ":" + getMinute() + ":" + getSecond();
                Text timeText = new Text(getW() * 0.4, getH() - 10, s);

                // Adding nodes to pane
                getChildren().clear();
                ObservableList<Node> list = getChildren();
                list.add(circle);
                Collections.addAll(list, dashes);
                Collections.addAll(list, texts);
                list.addAll(sLine, mLine, hLine, timeText);
        }

        public void start() {
                timeline.play();
        }

        public void stop() {
                timeline.pause();
        }

        private void update() {
                setCurrentTime();
                paintClock();
        }

        private Point2D getPointBCloserToA(Point2D a, Point2D b, double coefficient) {

                double deltaX = b.getX() - a.getX();
                double deltaY = b.getY() - a.getY();

                return new Point2D(a.getX() + coefficient * deltaX, a.getY() + coefficient * deltaY);
        }
}
```

### 2. Exercise_32.java

```
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
```

```
public class Exercise_32 extends Application {

	@Override
	public void start(Stage primaryStage) throws Exception {
		ClockPane clockPane = new ClockPane();
		clockPane.start();

		Button btStart = new Button("Start");
		btStart.setOnAction(e -> clockPane.start());
		Button btStop = new Button("Stop");
		btStop.setOnAction(e -> clockPane.stop());

		HBox hBox = new HBox(btStart, btStop);
		hBox.setSpacing(10);
		hBox.setPadding(new Insets(10, 10, 10, 10));
		hBox.setAlignment(Pos.CENTER);
		BorderPane borderPane = new BorderPane(clockPane, null, null, hBox, null);

		primaryStage.setScene(new Scene(borderPane));
		primaryStage.setTitle("Clock");
		primaryStage.show();
	}

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