1. Main Idea

To draw detail of my clock, I should add Clock scale (drawing scale) which made up of 48 small scale, and 12 big scale.

- ⇨ I implemented this by using for loop, and draw 4 small scale, and 1 big scale … so on. I calculated angle by using radian, sin, cos.

And then, I added 12 time (number) next to the big Scale.

- ⇨ I also implemented this by using for loop. While i increases from 1 to 12, calculate X coordinate and Y coordinate using radian, sin, cos.

Finally, I added total time, under my clock.

- ⇨ I calculated Y coordinate using centerY and circleRadius. X coordinate is same to centerX.

And here is my code below.

1. ClockAnimation.java

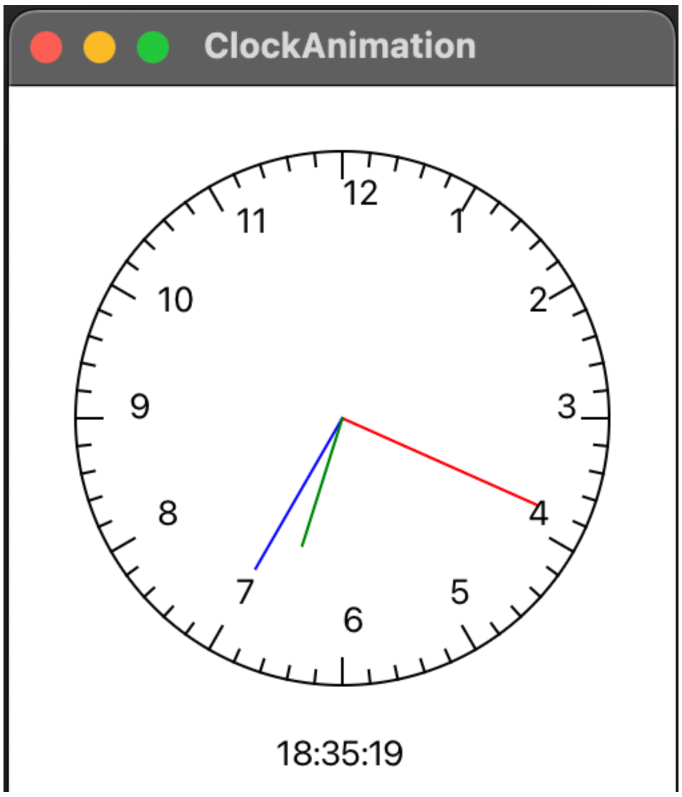
```java
package com.example.demo;

import com.example.demo.ClockPane;
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.util.Duration;


public class ClockAnimation extends Application{
    @Override
    public void start(Stage primaryStage){
        ClockPane clock = new ClockPane();

        EventHandler<ActionEvent> eventHandler = e -> {
            clock.setCurrentTime();
        };

        Timeline animation = new Timeline(new
KeyFrame(Duration.millis(1000), eventHandler));
        animation.setCycleCount(Timeline.INDEFINITE);
        animation.play();

        Scene scene = new Scene(clock, 250, 50);
        primaryStage.setTitle("ClockAnimation");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
}
```

This is exactly same to sample code. This java class makes new clockPane class, and clock animation so that clock can change each second.

2. ClockPane.java

```java
package com.example.demo;


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

public class ClockPane extends Pane {
    private int hour;
    private int minute;
    private int second;

    private double w = 250, h = 250;

    public ClockPane() {
        setCurrentTime();
    }

    public ClockPane(int hour, int minute, int second) {
        this.hour = hour;
        this.minute = minute;
        this.second = second;
        paintClock();
    }

    public int getHour() {
        return hour;
    }

    public void setHour(int hour) {
        this.hour = hour;
        paintClock();
    }

    public int getMinute() {
        return minute;
    }

    public void setMinute(int minute) {
        this.minute = minute;
        paintClock();
    }

    public int getSecond() {
        return second;
    }

    public void setSecond(int second) {
        this.second = second;
        paintClock();
    }

    public double getW() {
        return w;
    }

    public void setW(double w) {
        this.w = w;
        paintClock();
    }

    public double getH() {
        return h;
    }

    public void setH(double h) {
        this.h = h;
        paintClock();
    }

    public void setCurrentTime() {
        Calendar calendar = new GregorianCalendar();

        this.hour = calendar.get(Calendar.HOUR_OF_DAY);
        this.minute = calendar.get(Calendar.MINUTE);
        this.second = calendar.get(Calendar.SECOND);

        paintClock();
    }
    protected void paintClock() {
        double clockRadius = Math.min(w, h) * 0.8 * 0.5;
        double centerX = w / 2;
        double centerY = h / 2;

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


        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);

        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);

        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);

        getChildren().clear();
        getChildren().addAll(circle, sLine, mLine, hLine);


        for (int i = 1; i <= 12; i++){  // add clock time
using sLength(second length) and center Coordinate with cos,
sin
            getChildren().add(new Text(centerX + sLength *
Math.sin((2 * Math.PI / 12) * i), centerY - sLength *
Math.cos((2 * Math.PI / 12) * i), Integer.toString(i)));
        }

        for (int i = 1; i <= 60; i++){  // add clock scale
            double scaleX = centerX + clockRadius * Math.sin(2
* Math.PI / 60 * i); // scale X starting point
            double scaleY = centerY - clockRadius * Math.cos(2
* Math.PI / 60 * i); // scale Y starting point

            double scaleLength = clockRadius * 0.95;    //
small scale's length is 0.05 * clock Radius
            if (i % 5 == 0){
                scaleLength = clockRadius * 0.9;    // big
scale's length is 0.1 * clock Radius
            }

            double scaleX2 = centerX + scaleLength *
Math.sin(2 * Math.PI / 60 * i);    // scale X ending point
            double scaleY2 = centerY - scaleLength *
Math.cos(2 * Math.PI / 60 * i);    // scale Y ending point

            getChildren().add(new Line(scaleX, scaleY,
scaleX2, scaleY2));  // add scale Line
        }

        // add total time under my clock : first make time
string, and add using center coordinate and clockRadius
        String currentTime = Integer.toString(this.hour) + ":"
+ Integer.toString(this.minute) + ":" +
Integer.toString(this.second);
        getChildren().add(new Text(centerX -25, centerY +
(clockRadius + 30), currentTime));
    }
}
```

3. Implemented result


ClockAnimation
12
1
2
3
4
5
6
7
8
9
10
11
18:35:19