Langara
THE COLLEGE OF HIGHER LEARNING.

Department of Computing Science & Information Systems
CPSC 1181
Lab#7
Oct 27, 2022
Objectives:

Event Handling
Layout management

Preparation:

Study class notes and exercised

Due date:

Due Date: 11:00 PM on Wednesday Nov 2, 2022

Where to upload:

Zip your files into yourstudentID.zip where yourstudentID is your student number, and upload it to dropbox lab7 in D2L.

Part A: Tutorial

No tutorial this week

Part B:

Important Notes:

1. It is important to follow the methods taught in the slides for JavaFX. If you stray from them, you will likely receive significant penalties or even zero. If there is something you'd like to try and are unsure if it is allowed, contact your instructor.

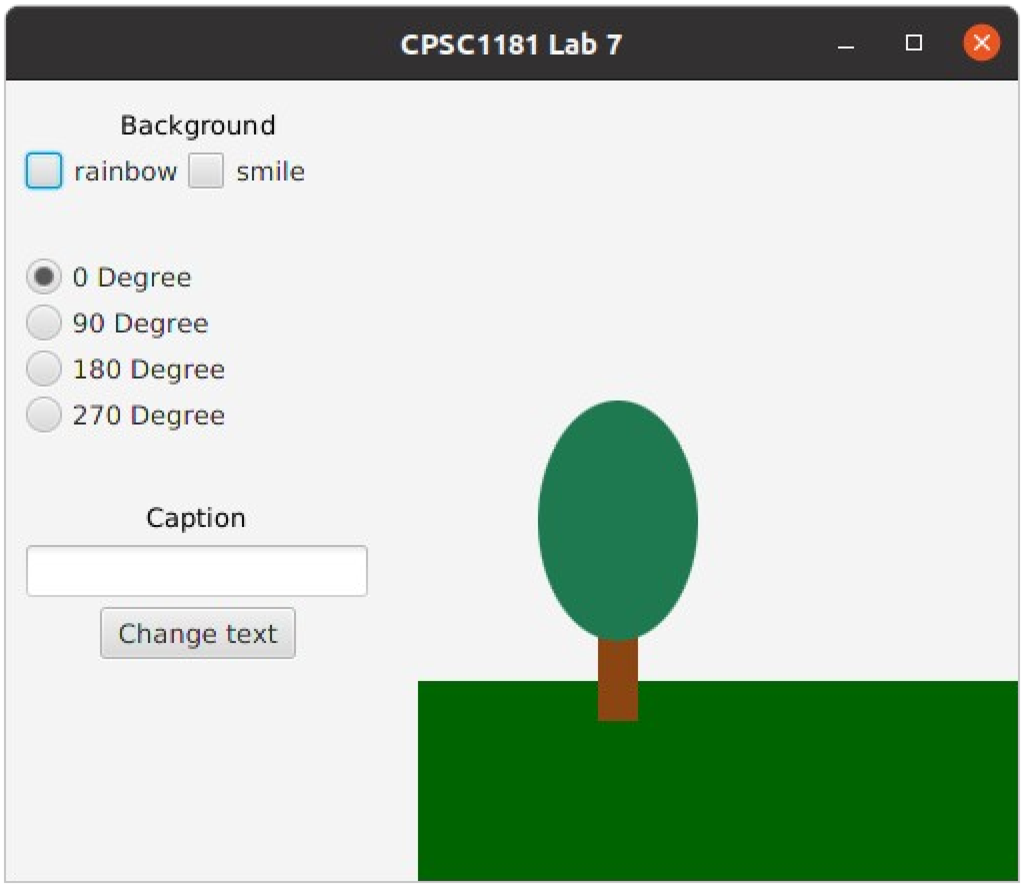
2. You are not allowed to use setLayoutX(...) or setLayoutY(..) to manage elements on the pane. You should use appropriate layout management to do it, otherwise, you may get poor mark for this lab assignment.

3. You must use Inner classes as shown in the notes. If you use ***Lamda*** expression or ***Anonymous classes*** you will get zero mark for this lab assignment.

Use HBox, VBox, BorderPane

What to do:

You are going to build a JavaFX Application based on the examples we use in our notes.
(or a similar alternative you've created yourself) as a starting point.
Place the background, grass and the grass and tree on a Pane that will be placed in the centre area of a BorderPane.



Check the demo.
Similar to the Message example from the lectures. Checkboxes, radio buttons, a textbox, and button will change what is displayed in the centre pane.

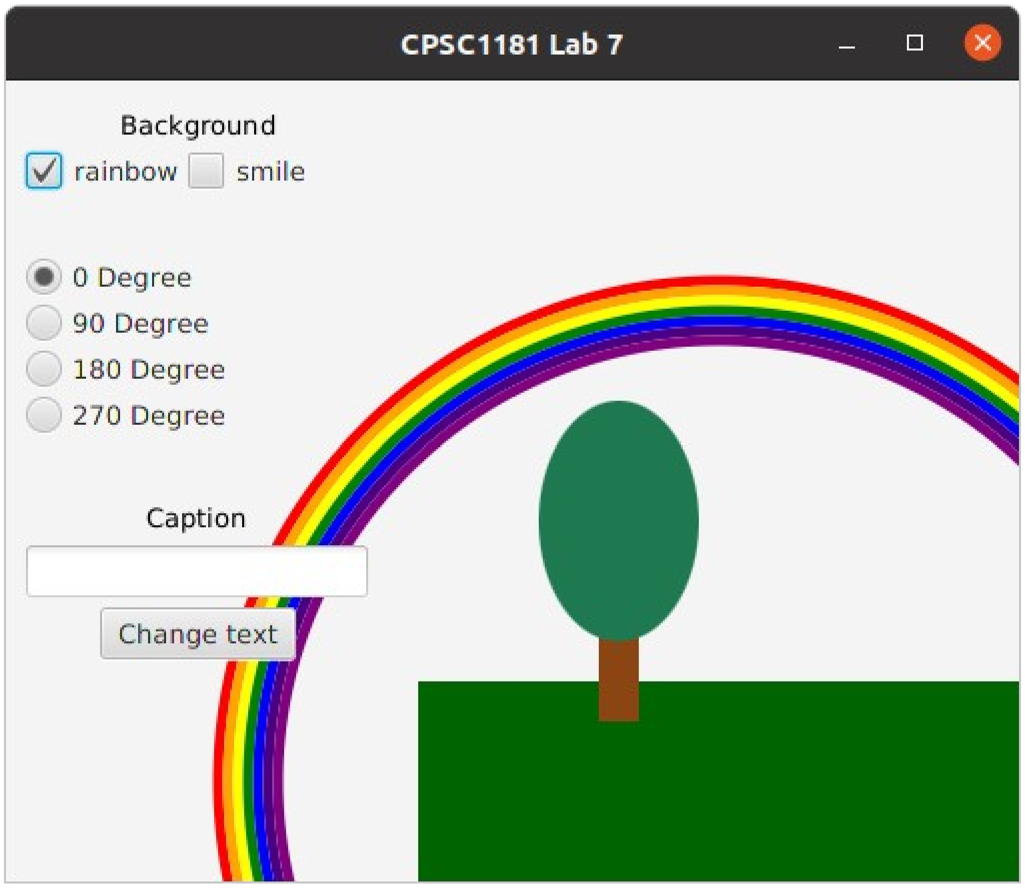
The application must include

- When clicked they hide/display elements from the background (don't worry whether an element is truly background or not. As long as an element appears and disappears, it is ok)
- A set of four radio buttons that control the rotation of the tree on the screen.
- A textbox and button that allow you to display text somewhere within the graphics. The text can be positioned anywhere, but must have a color/font that makes it easy to read.
- Elements can be hidden/visible/rotated when your Application starts, as long as the graphics match the state of the checkboxes/radio buttons. For example, if the checkbox is checked, the element should be displayed.

Hints:

- You can just rename the root Pane that you are drawing everything on. Then make a BorderPane the new root. Then add the original Pane to the centre area of the BorderPane. Give that Pane the correct size for your graphics.
- Set up the rest of the elements in the layout. Use only HBoxes, VBoxes, and alignment/margins/padding to position everything.
- You don't need to match my example, but you should produce something similar. Your program should:
    - Use the same basic structure with similar clusters on the left side. You must have pairs of checkboxes and radio buttons.
    - The basic alignment (left/centre/right) of elements should be the same
    - There should be padding/margin used to create separation from the edge of the window and between elements in the window
- The left area contains a VBox containing a Text element ("background"), an HBox (check boxes), a VBox (radio buttons), a Text element (Caption), and the textfield and a button.

Please Read the following Notes before you start



Solving Clipping (only if necessary)
If your graphics goes beyond the edge of the Pane, they would have been cut off naturally. But with the BorderPane layout, they will bleed into the other areas of the BorderPane like the rainbow in the image to the right. But, this is easy to solve.
You need to create a Rectangle that is positioned at 0,0 and the same dimensions as the pane for your graphics. Then use the clip method. Then everything will be cut (clipped) off at the edge of the pane.

```
treePane = new Pane();
treePane.setPrefWidth(300);
treePane.setPrefHeight(400);
Rectangle clip = new Rectangle(0,0,300,400);
treePane.setClip(clip);
```

Make Everything Work By Adding Event Handlers

Now, create event handler inner classes for the needed operations. See the following sections for help on making things hidden/visible or rotated

- Don't create a separate event handler for every checkbox, radio button, and other element, but don't just create a single event handler called by everything. I used three event handler classes for my application.

- The change text button should update a Text element you have in the pane with the graphics. If the Text element is displaying the empty string, nothing will be displayed.

Grouping Shapes Together

To simplify the code, it is best to add all of the graphical shapes of the elements being changed to a Group object

- Create a Group object. Then add all shapes that make up an element to one group.
- Then instead of adding the shapes to the pane, add the shapes to the Group and add the Group to the pane.
- Now you can use the Group to more simply hide/show or rotate an element.
- You can do this with the start method or within an inner class for an element

Here is example code for the Smile in the example.

```
class Tree {
        private Group gr;
        private final int WTRUNK = 20;
        private final int HTRUNK = 100;
        private final int WLEAVES = 40;
        private final int HLEAVES = 60;
        public Tree(int x, int y) {
                gr = new Group();
                Rectangle trunk = new Rectangle(x-WTRUNK/2, y-HTRUNK, WTRUNK, HTRUNK);
                trunk.setFill(Color.SADDLEBROWN);

                Ellipse leaves = new Ellipse(x,y-HTRUNK,WLEAVES,HLEAVES);
                leaves.setFill(Color.rgb(30,120,80));

                gr.getChildren().addAll(trunk,leaves);

        }
        // returns group
        public Group getNodes(){
                return gr;
        }
}
```

Hiding/Showing or Rotating Elements
Once you have the groups set up, this is simple.

- To make a group hidden/visible use .setVisible(boolan visible)
    - smilySun.setVisible(true) makes the smile appear
- To rotate a group, use setRotate(double angle)
    - gr.setRotate(0) sets it upright and gr.setRotate(90) turns it 90 degrees (sideways)
    - The group will automatically rotate around its center-point

What to submit

1. Comment all your classes and methods using the javadoc notation.
2. Your source files.
3. Comments about your assignment if needed these comments are not the comments documenting your code but rather something you need to convey to us about your assignment
4. Submit to D2l Dropbox: lab7

TOTAL MARK: 70