CS112 – Spring 2024
Lab27
Instructor: Paul Haskell

INTRODUCTION

In this lab, you will extend your LavaLamp from Monday, to make it groovier than it was :)

LavaLamp

Your **LavaLamp2.java** program shall create a `JPanel` that contains a single `JButton`. Your program also shall create a `Timer` and a background `Color`. Every 100 milliseconds, the handler for `Timer` events should update the background color of the `JPanel`. The updates should be **small** random adjustments (i.e. small changes) to the Red, Green, and Blue components of the `JPanel`'s background color. Values for each color component should be restricted to the range [0, 255]. You can experiment with how to update the color components to give you pleasing-looking small color changes.

When the `JButton` is pressed once, the colors should stop updating and the current red/green/blue color components should be printed to `System.out` as three numbers separated by commas, e.g.

```
112,87,34
```

When the `JButton` is pressed again, the colors should start changing again. This alternating behavior should continue.

- Third button press stops the colors from changing and prints current color components
- Fourth button press restarts the colors changing
- etc

Ok, now for the second part! Make a new "main" `JPanel` that you insert into the `JFrame`. Give this `JPanel` a preferred size of 700 x 500, and set its background color to something you find attractive. Now insert your color-changing `JPanel` into this "main" `JPanel`. Here are some details:

- Give the color-changing `JPanel` a size of 300 x 200. You must use the `setSize()` method, not `setPreferredSize()`. The `setPreferredSize()` method is used only by "top-level" `JPanels`.

- Remember the discussion in Lecture 27 about Layout Managers? For this lab, you must disable the default layout manager. On the "main" `JPanel` object, call:
  ```
  setLayout(null)
  ```
  to disable the layout manager.
- You can set the position of the color-changing `JPanel` inside the "main" `JPanel` using an object called a `Point` (which has x-y coordinates) and the `setLocation()` method. For example:
  ```
  Point p = new Point(150, 100);
  colorChangingJPanel.setLocation(p);
  ```
- Did that work? Now, just as you make small random changes to the color, also make small random changes to the color-changing `JPanel's` **position**. You must keep the color-changing `JPanel's` position inside the "main" `JPanel`!

That's it! Once you get this working, you have a real lava lamp.

Reminder

Put all your files in your **Lab27** directory and push to GitHub before the deadline. This assignment must be turned in before the end of class today.

Conclusion

This lab is simply intended to give you some useful graphics software of your own, and something groovy to run in the background while you continue to work on your project.

Grading Rubric

LavaLamp2.java is worth 25 points. 5 points if it compiles, 10 points if it operates correctly (changing colors slowly and randomly, printing colors, button control), and 0-10 points based on the grader's subjective judgment of code, design quality, and grooviness (i.e. small but random changes to color and position).