JavaFX Paint

Tyler Pennington

1 Problem Description

In this homework you are going to use JavaFX to create a paint tool similar to Microsoft Paint. This assignment will test your understanding and ability to use lambda expressions and event-driven programming.

You will be provided one file, `Tool.java`. `Tool` is an interface that represents a general tool used for drawing on the canvas, such as a Pencil tool or Rectangle tool. `Tool` has three abstract methods: `onPress`, `onDrag`, and `onRelease`. These methods represent actions that should occur when the mouse button is pressed down, dragged while still pressed down, and then released. You will implement this interface to create different tools for your paint program.

2 Requirements

This assignment is meant to be open-ended. The implementation and design is up to you to decide. **The quality of your design will be graded in accordance with the concepts you have learned in this class.**

At the very minimum, your submission should include the following:

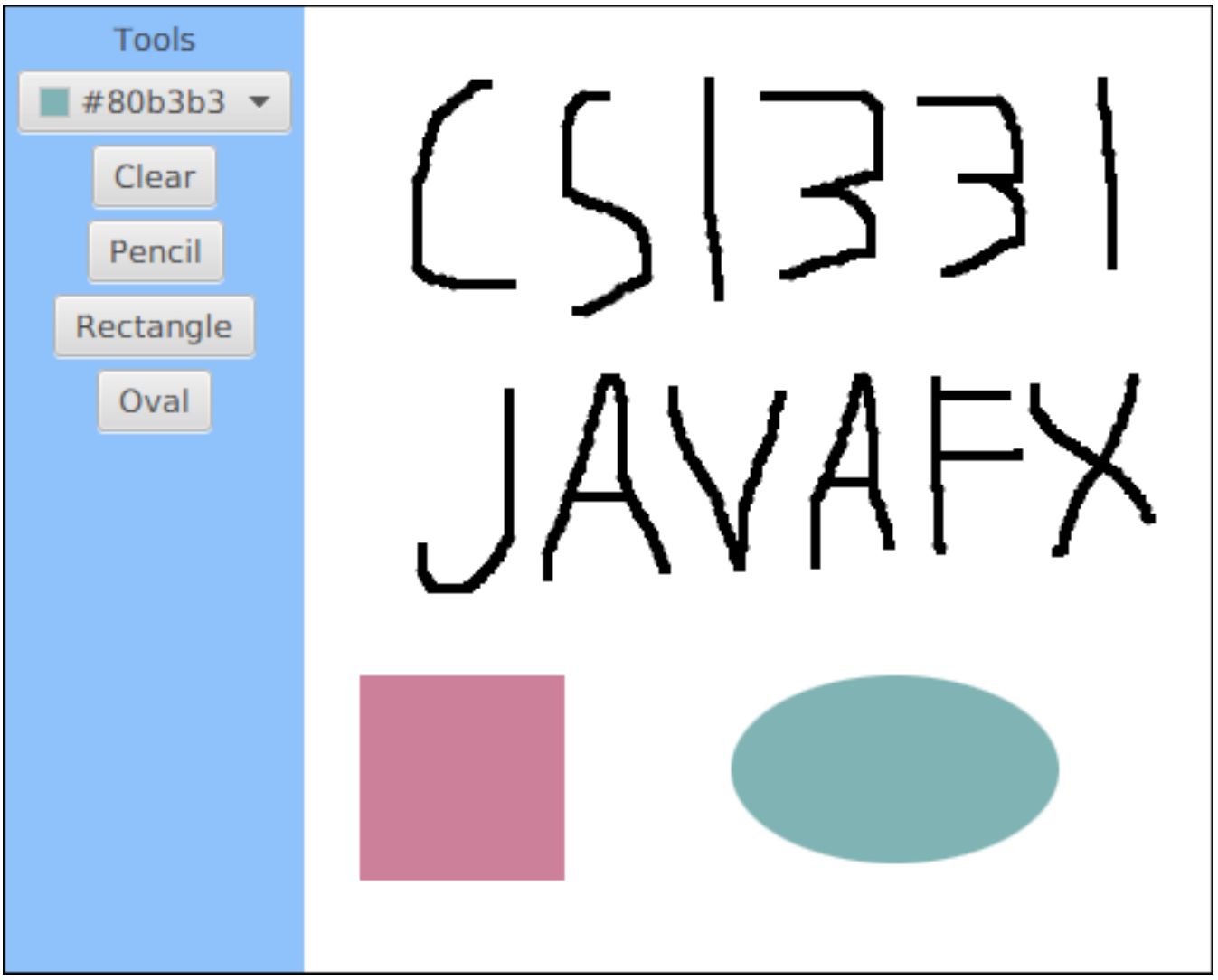
1. A file named `PaintFX.java` which starts up your JavaFX application.
2. A canvas that you can draw on using various tools, and a way to clear everything on the canvas.
3. A way to change the current color.
4. You will also need to implement the following tools:
   (a) A Pencil tool to draw simple lines on the canvas.
   (b) A Rectangle tool to draw solid rectangles.
   (c) One (or more) tools of your choosing. This can be another kind of shape, an eraser, a paintbrush, or anything else that could be considered useful. Try to have fun with this one, but please don't submit something trivial or something that you wouldn't normally see in a paint program. Feel free to add as much functionality as you want as long as the base requirements are fufilled.

3 Solution and Tips

To help you with this assignment, here are some suggested readings and links to useful classes in the JavaFX API.

1. JavaFX Layouts
2. Helpful JavaFX Classes: `Canvas`, `GraphicsContext`, `ColorPicker`, `Color`
3. Re-read the lecture slides on action events in JavaFX. You will need them to complete this assignment.

4 Example



5 Javadocs

We are going to have you do Javadocs for this assignment (and for all assignments here on out). Javadocs are a clean and useful way to document your code's functionality. For more information on what Javadocs are and why they are awesome, the online documentation for them is very detailed and helpful. The relevant tags that you need to have are `@author`, `@version`, `@param`, and `@return`. Here is an example of a properly Javadoc'd class:

```
import java.util.Scanner;

/**
 * This class represents a Dog object.
 * @author George P. Burdell
 * @version 13.31
 */
public class Dog {

   /**
    * Creates an awesome dog (NOT a dawg!)
    */
```

```
    public Dog(){
        ...
    }

    /**
     * This method takes in two ints and returns their sum
     * @param a first number
     * @param b second number
     * @return sum of a and b
     */
    public int add(int a, int b){
        ...
    }
}
```

6 Checkstyle

You must run checkstyle on your submission. The checkstyle cap for this assignment is **100** points.
Review the Style Guide and download the Checkstyle jar and associated XML file. Run Checkstyle on your code like so:

```
$ java -jar checkstyle-6.0-all.jar -c cs1331-checkstyle.xml *.java
Starting audit...
Audit done.
```

The message above means there were no Checkstyle errors. You can easily count Checkstyle errors by piping the output of Checkstyle through `wc -l` and subtracting 2 for the two non-error lines printed above (which is how we will deduct points). For example:

```
$ java -jar checkstyle-6.0-all.jar -c cs1331-checkstyle.xml *.java | wc -l
      2
```

Alternatively, if you are on Windows, you can use the following instead:

```
C:\> java -jar checkstyle-6.0-all.jar -c cs1331-checkstyle.xml *.java | findstr /v "Starting audit..." |
     findstr /v "Audit done" | find /c /v "hashcode()"
0
```

> Food for thought: is there a one-liner like above that shows you only the number of errors? Hint: `man grep`.

The Java source files we provide contain no Checkstyle errors. For this assignment, there will be a maximum of **100** points lost due to Checkstyle errors (1 point per error). In future homeworks we will be increasing this cap, so get into the habit of fixing these style errors early!

7 Turn-in Procedure

Submit all of the Java source files you modified and resources your program requires to run to T-Square. Do not submit any compiled bytecode (`.class` files), the Checkstyle jar file, or the `cs1331-checkstyle.xml` file. When you're ready, double-check that you have submitted and not just saved a draft.

Please remember to run your code with Checkstyle!

Verify the Success of Your Submission to T-Square
Practice safe submission! Verify that your HW files were truly submitted correctly, the upload was successful, and that the files compile and run. It is solely your responsibility to turn in your homework and practice this safe submission safeguard.

1. After uploading the files to T-Square you should receive an email from T-Square listing the names of the files that were uploaded and received. If you do not get the confirmation email almost immediately, something is wrong with your HW submission and/or your email. Even receiving the email does not guarantee that you turned in exactly what you intended.
2. After submitting the files to T-Square, return to the Assignment menu option and this homework. It should show the submitted files.
3. Download copies of your submitted files from the T-Square Assignment page placing them in a new folder.
4. Recompile and test those exact files.
5. This helps guard against a few things.
   (a) It helps insure that you turn in the correct files.
   (b) It helps you realize if you omit a file or files. [1] (If you do discover that you omitted a file, submit all of your files again, not just the missing one.)
   (c) Helps find last minute causes of files not compiling and/or running.

[1] Missing files will not be given any credit, and non-compiling homework solutions will receive few to zero points. Also recall that late homework will not be accepted regardless of excuse. Treat the due date with respect. The real due date is midnight. Do not wait until the last minute!