CS 2110 — Fall 2012

Homework 4

Paint Program

Due: Wednesday, 12 November, 11:59PM

In this assignment, you will write parts of a simple paint program. Some of the functionality you will implement is:

1. Freehand drawing, erasing, airbrushing.
2. Drawing lines and circles.

0 Instructions

0.1 Grading, Partners, Academic Integrity, Help

Solutions will be graded on correctness, the quality of the algorithms, and style. A correct program compiles without errors or warnings and behaves according the the requirements given here and in the comments of the code. A program with good style is clear, concise, and easy to read.

You can work in groups of two. Form the group well before the assignment due date. Both must do something to form the group: one proposes, the other accepts. People in a group must work together. It is against the rules for one person to do programming on this assignment without the other person sitting nearby and helping. Take turns `driving` —using the keyboard and mouse.

With the exception of your CMS-registered group partner, you may not look at anyone else's code, in any form, or show your code to anyone else (except the course staff), in any form. You may not show or give your code to another student in the class.

We will pin a note on the Piazza, as we did for A4, to discuss important points. Check it often.

1 Installing the code

The release contains source code (files with extension `.java`) and image files (with extension `.png`). Follow the following steps to install the code:

1. Create an empty Java project in eclipse and copy the code into the source (`src`) folder of the project. The easiest way to do this is by dragging the code to the appropriate place inside Eclipse.
2. Copy the image files to the default current directory. This is typically the project directory (inside the workspace folder, there is a folder with the same name as the Eclipse project, and this is the project directory).

Method main for the program is in file `PaintGUI.java`.

2 Example of drawing in Java

We give you a simple example of drawing in Java. Suppose you want to create an image object, e.g. a rectangular image of width 800 pixels and height 600 pixels. You want to color the entire image with a single color. The main steps involved are the following:

- Construct the image object:

  ```
  BufferedImage img= new BufferedImage(width,height,BufferedImage.TYPE_INT_ARGB);
  ```

  The argument `BufferedImage.TYPE_INT_ARGB` tells Java to create an image with 8-bit RGBA (red, green, blue, alpha) color components packed into integer pixels. The alpha component corresponds to transparency.

- Obtain the "graphics" object of the image, which exposes drawing functionality.

  ```
  Graphics2D g2d= (Graphics2D) img.getGraphics();
  ```

- Set the drawing color.

  ```
  g2d.setColor(drawingColor);
  ```

- Fill the rectangular area with top-left corner (0,0) with the current drawing color.

  ```
  g2d.fillRect(0, 0, width, height);
  ```

You can also explicitly construct shapes, like rectangles

```
Shape rect= new Rectangle2D.Double(topLeftX, topLeftY, w, h);
```

and draw them on the image (the outline of the shape)

```
g2d.draw(rect);
```

or draw them filled with the current drawing color

```
g2d.fill(rect);
```

For more information, you can read Java's tutorial:

`http://docs.oracle.com/javase/tutorial/2d/index.html`

The code we have provided contains several examples that are intuitive enough to understand without going into too much detail in Java's vast API for 2D graphics.

3 Paint program

You can see a screenshot of a fully implemented paint program in Figure 1, where we also show it annotated with what the important GUI components represent.

- The program allows you to open image files and save in images files. See Figure 2.
- You can draw with a pencil (freehand drawing), erase, airbrush. You can draw lines and circles. You can change both the foreground color (used for drawing) and the background color (used for erasing). See Figure 3.

  SAVE: The program lets you know when there are unsaved changes to the image. Label "SAVE" appears at the bottom whenever changes are made and disappears when the image is saved to a file.

You will see some more functionality by running the release code.

4 Project Structure

A significant part of your work for this assignment will be to read the release code and figure out how it works. You may need to find the relevant Java documentation on the Web and read it. Successful completion of the assignment requires you to understand well how the existing code works. Here is a brief description of classes:

`DrawingCanvas`**:** This class represents the image you are painting. It handles the mouse events that are relevant for drawing and performs the drawing operations.

`PaintGUI`**:** This class sets up the main window of the program. It handles the creation of most GUI components and responds to relevant actions.

`NewImageDialog`**:** A custom dialog for creating new blank images.

`Tool`**:** An enumeration of the available drawing tools.

`DrawingCanvas` and `PaintGUI` are the only classes you need to modify. **Do not change anything else in the code**. Every point inside the code where you have to change/add something has been marked for you.

5 Class `DrawingCanvas`

The parts that have to be implemented have been clearly marked in the release code. You will find detailed instructions in the code. Some remarks:

- Freehand pencil drawing should leave no gaps. That is, when you press and drag the mouse around, it should draw on the trace of the mouse movement. The eraser should behave similarly.

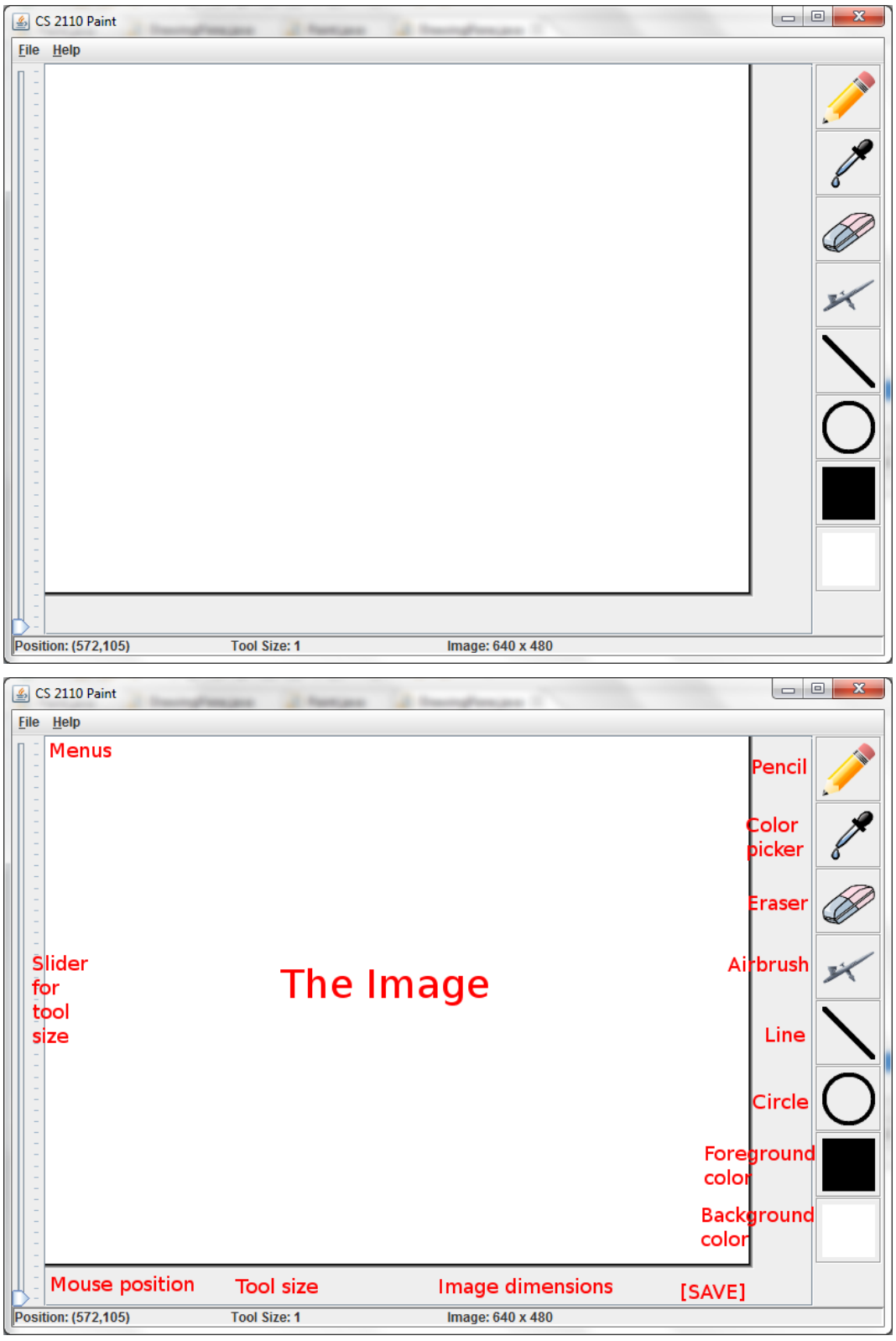

Figure 1: Fully implemented paint program.

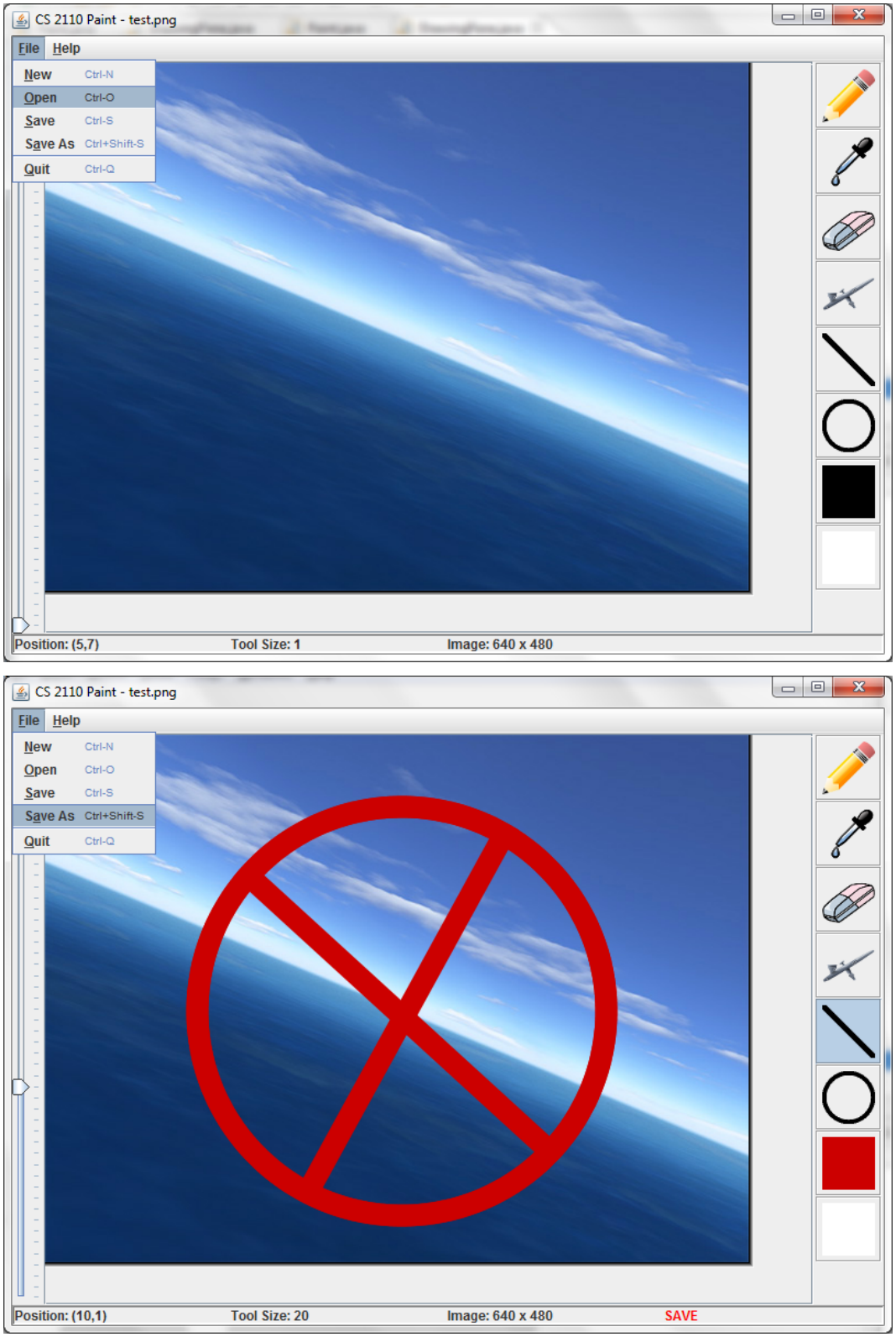

Figure 2: Open and save image files.

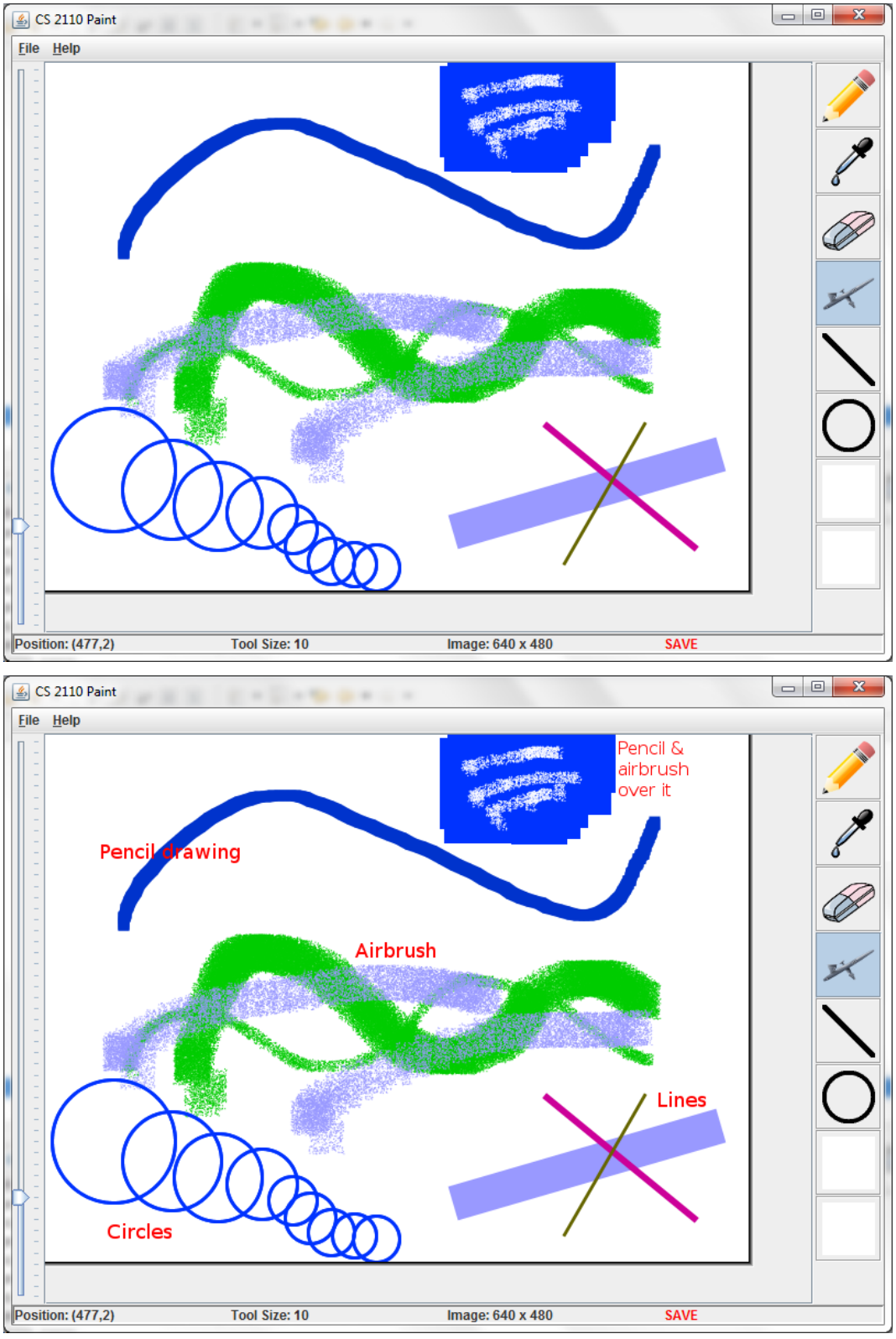

Figure 3: Drawing.

- The airbrush does not have to leave a continuous trace.
- We suggest you implement line drawing as follows: When you first press the mouse, one endpoint of the line is fixed (nothing is drawn on the image yet). When you press the mouse for a second time, the second endpoint is fixed and the line is drawn on the image.

  You should visualize the "tentative line" after the first endpoint is given and before the second endpoint is given. That is, while the user is deciding on the second endpoint and moving the mouse around, they can see the line that will be drawn.
- Do something similar for the circle. For example, when you first press the mouse, you fix the center of the circle and the second time you fix a point that is on the circle and draw.

6 Class `PaintGUI`

The parts that have to be implemented have been clearly marked in the release code. There is very little to be done here and it is very straightforward, but you need to read and understand the code well before you attempt to make any changes.

7 Submission

Compress exactly the following files into a `zip` file that you will then submit on CMS:

- `README.txt`: This file should contain your name, your NetID, all known issues you have with your submitted code, and the names of anyone you have discussed the homework with (except the course staff). Also, if you want to explain something about your code that you think needs clarification, you can add a few paragraphs here.
- All the `.java` files needed for your program.

Do not include any files ending in `.class`.

All `.java` files should compile and conform to the prototypes we gave you. We write our own classes that use your classes' public methods to test your code. *Even if you do not use a method we require, you should still implement it for our use.*