**CS 342 Report for (Lab 2)**

**Code**

// Programmer: Bradley Golden

// Assignment: Lab 2

// Date: September 6, 2015

// Description: A basic front-end applet with the ability to display a rectangle or message

import java.awt.FlowLayout;

import java.awt.Graphics;

import java.awt.event.\*;

import javax.swing.\*;

import java.applet.\*;

public class Lab2 extends JApplet implements ActionListener

{

private JPanel contentPanel; // panel to hold contents

// row 1 contents

private ButtonGroup squareOrMessageGroup; // groups draw sqaure or write message elements

private JRadioButton drawSquareButton; // radio choice for drawing square

private JRadioButton writeMessageButton; // radio choice for writing a message

private JTextField messageField; // message field for inputting a message

// row 2 contents

private JLabel selectLabel; // label to descript combobox

private static final String[] CHOICES = {" ", "Center", "Random"}; // user choices

private JComboBox selectBox; // providest the user with a set of options from CHOICES

private JCheckBox drawColor; // option to draw a rectangle/message in color

private JButton drawIt; // initiate the drawing

@Override

public void init()

{

/\*

\* Row 1 Contents

\*/

// initialize row 1 contents

drawSquareButton = new JRadioButton("Draw square", true); // initialize draw square radio

writeMessageButton = new JRadioButton("Write message:", false); // initialize write message radio

messageField = new JTextField(20); // initialize message field

// add contents to panel

contentPanel = new JPanel();

contentPanel.add(drawSquareButton);

contentPanel.add(writeMessageButton);

contentPanel.add(messageField);

// logically link radio buttons

squareOrMessageGroup = new ButtonGroup();

squareOrMessageGroup.add(drawSquareButton);

squareOrMessageGroup.add(writeMessageButton);

/\*

\* Row 2 Contents

\*/

//initialize row 2 contents

selectLabel = new JLabel("Select where to draw:"); // initialize elements

selectBox = new JComboBox(CHOICES);

selectBox.setMaximumRowCount(3);

drawColor = new JCheckBox("Draw in color");

drawIt = new JButton("Draw It!");

// add row 2 contents to panel

contentPanel.add(selectLabel);

contentPanel.add(selectBox);

contentPanel.add(drawColor);

contentPanel.add(drawIt);

add(contentPanel); // add panel contents to GUI

} // end init

@Override

public void paint(Graphics g)

{

super.paint(g);

} // end paint

public void actionPerformed(ActionEvent e)

{

repaint();

} // end actionPerformed

} // end Lab2 class

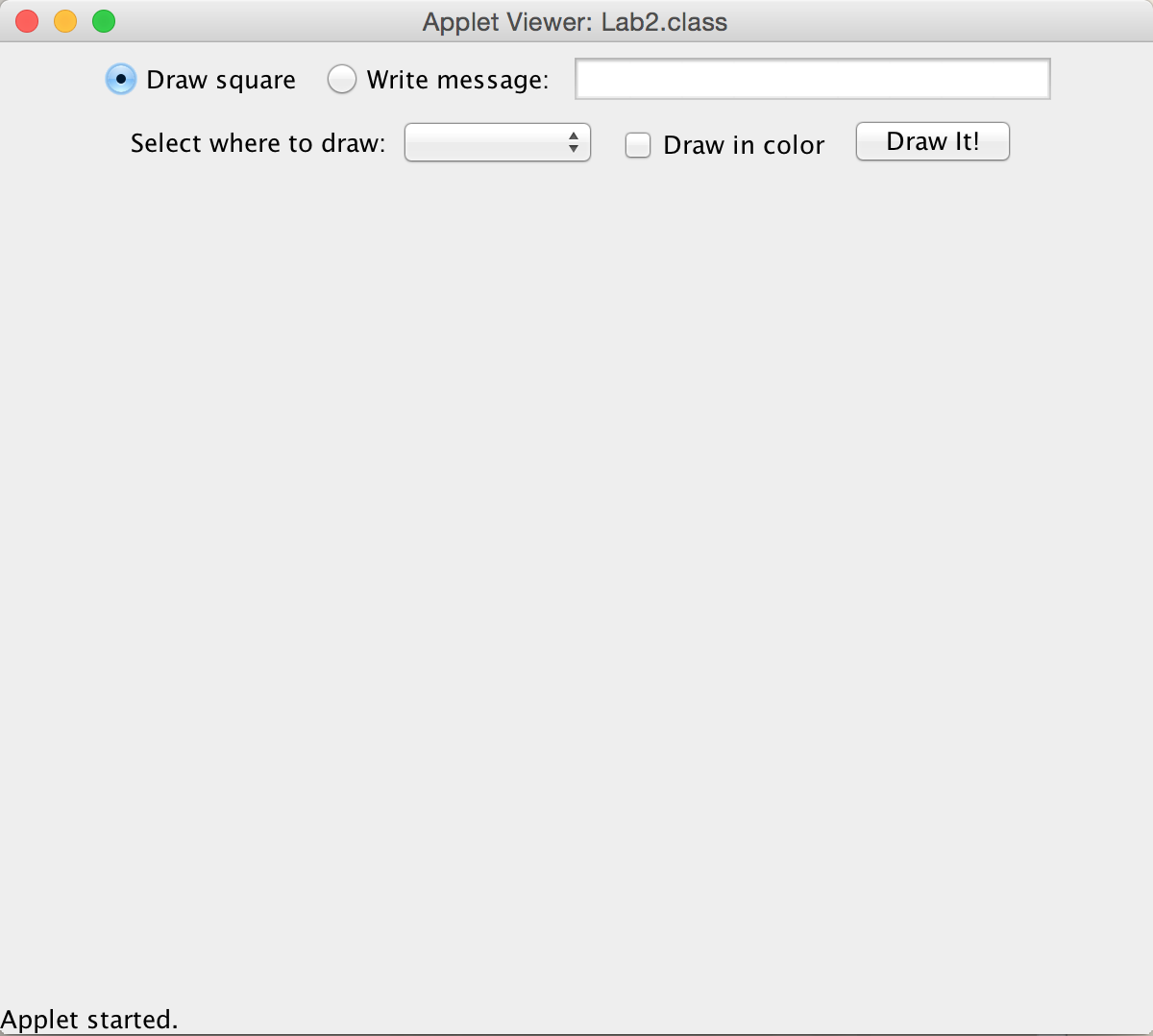
**User Interface**

<html>

<applet code="Lab2.class" width="600" height="500"></applet>

</html>

**Sample Runs**



**Discussion**

This was a brief introduction to the IPO model and the process of developing applications on the front-end. One major concept that I learned in this programming assignment is that the various layouts for GUI applets can be fairly complex. In the case of this program, I attempted to use a grid layout of 2 rows and 1 column. This did not work however. Instead, I put all window elements into a single panel and relied upon the window dimensions to “chop” runoff elements to the next row. This case would not be ideal if I decided to scale this project up or down because the GUI elements would not align properly in the window.