COMP1406 - Assignment #5

(Due: **Monday, February 24th @ 11:30pm)**

In this assignment you will create the "look" for two GUI applications. You WILL NOT be getting the applications to do anything. We are only interested in seeing how well you can arrange the GUI components onto the window. So, you DO NOT need to do any event handling.

---

(1) The **InfoApp** GUI

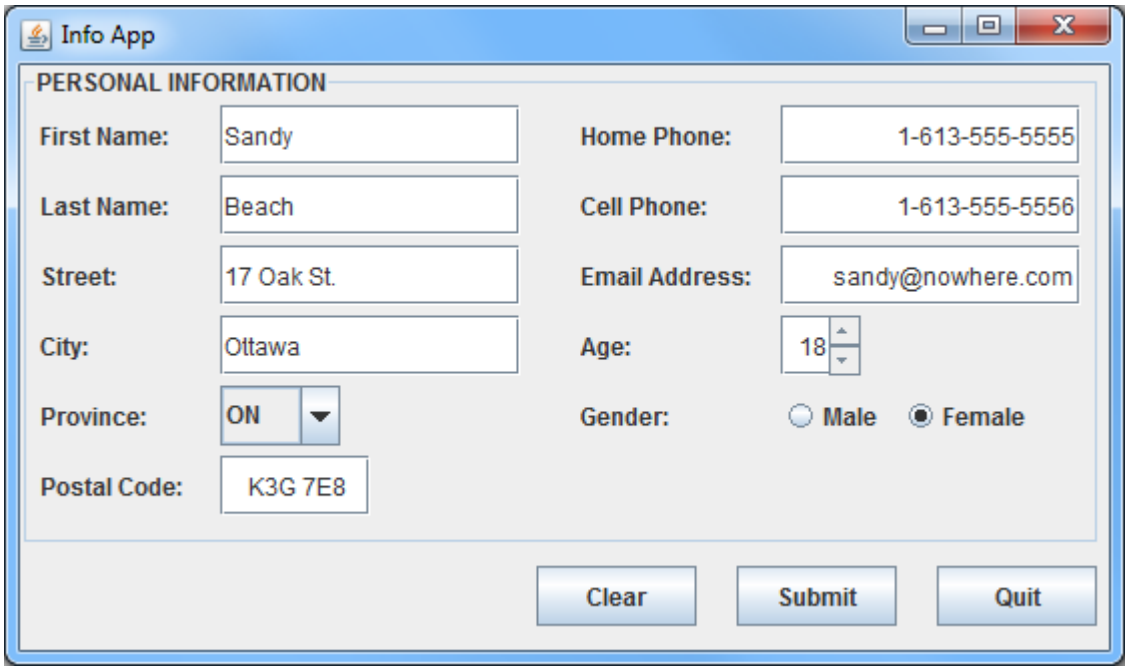
Create the following GUI. You MUST create a class called **InfoApp** to represent the main **JFrame** window. Also, the window contains a **JPanel** that represents all the personal information as well as 3 **JButtons** as shown. Make sure to set the Border label (see the example in the notes). This window shown here has a size of 560 x 330 pixels. You should NOT use any layout managers ... but MUST lay the components out manually by calculating the location and size of each component.

The **JLabels** and **JTextfields** below all have a height of 30 pixels. Note that the **Home Phone**, **Cell Phone** and **Email Address** text fields are all right-aligned (check the java documentation or google to see how to do this). The **Postal Code** field is center-aligned. The **Province** option must be a **JComboBox** with the following exact array data inside of it:

```
{"AB","BC","MB","NB","NL","NS","NT","NU","ON","PE","QC","SK","YT"}.
```

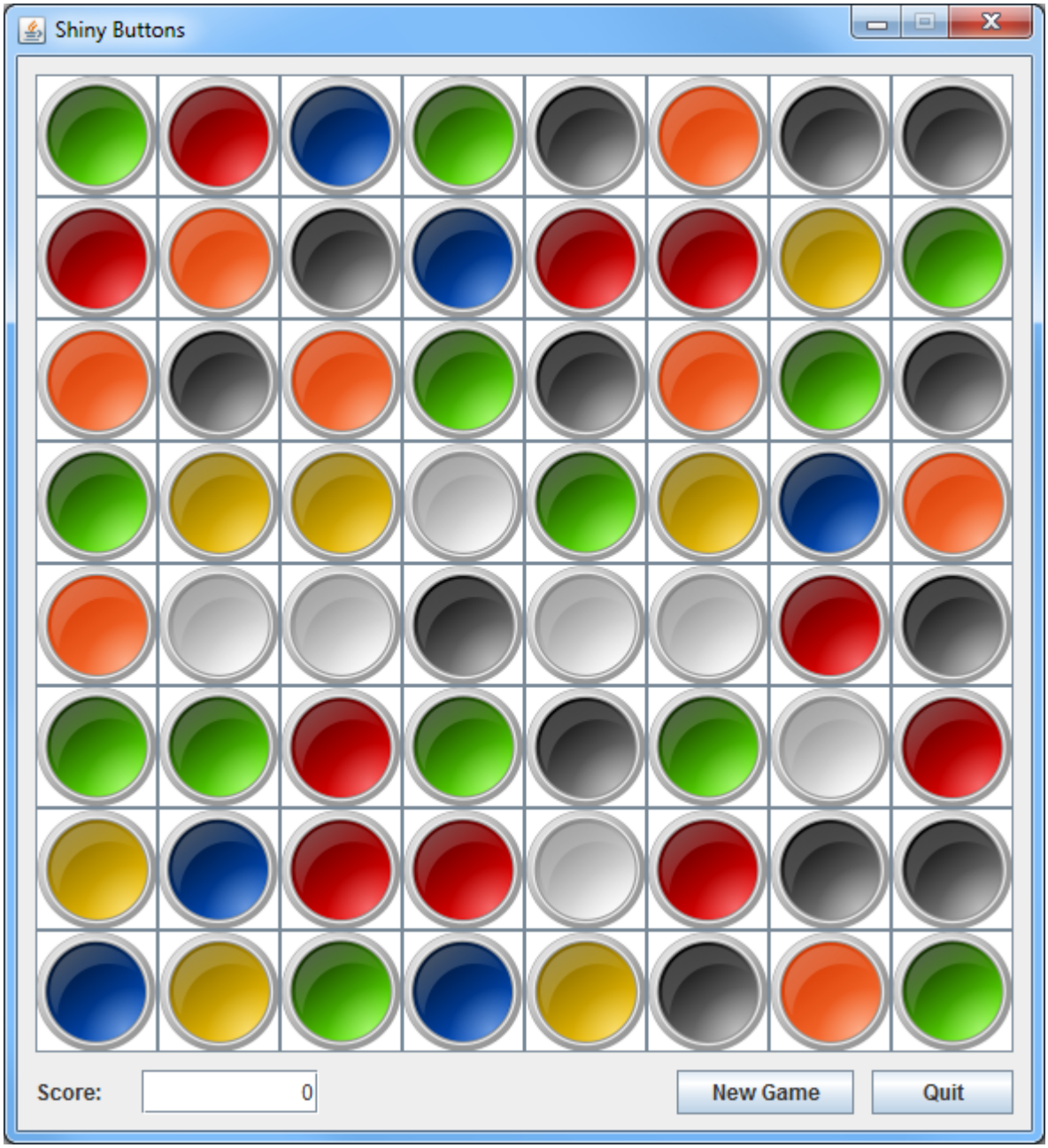
The **Age** option must be a **JSpinner** with an initial value of **18** (see java docs to find out how to set the **JSpinner** value). Finally, the **Gender** option must consist of two **JRadioButtons** which are placed into a **ButtonGroup** so that only one may be on at a time.

Note that you DO NOT have to get this interface to work. We are only interested in how you lay out the components and how they are formatted.



(2) The **ShinyButtons** GUI

Create the following GUI. You MUST create a class called **ShinyButtonsApp** to represent the main window. You WILL NOT have to implement this game, nor get any buttons working. You are ONLY being marked for the appearance and layout of the components on the window.



The window must contain a **JPanel** that contains an 8 x 8 array of **JButtons**, each with a size of 69 x 69 pixels and displaying an **ImageIcon** instead of text. At the bottom of the window is a **JLabel**, a right-aligned **JTextField** and two more **JButtons** as shown. The window must be non-resizable with a size of 578 x 634 pixels. Upon startup, the window should show a random arrangement of colored buttons as shown. You should make use of the following "model" class to represent the values for each button that will be displayed in the GUI. The class maintains a 2D array of bytes that represent color indices.

Note that you DO NOT have to get this interface to work. We are only interested in how you lay out the components and how they are formatted.

```
public class ShinyButtons {
    public static byte    RED = 0;
    public static byte    ORANGE = 1;
    public static byte    YELLOW = 2;
    public static byte    GREEN = 3;
    public static byte    BLUE = 4;
    public static byte    LIGHT_GRAY = 5;
    public static byte    DARK_GRAY = 6;

    public static byte    ROWS = 8;

    private byte[][]      buttonTable;

    public ShinyButtons() {
        buttonTable = new byte[ROWS][ROWS];
        resetButtons();
    }

    private void resetButtons() {
        for (int r=0; r<ROWS; r++)
            for (int c=0; c<ROWS; c++)
                buttonTable[r][c] = (byte)(Math.random()*7);
    }

    public byte getButton(int r, int c) { return buttonTable[r][c]; }
}
```

In your GUI, you may make use of the following array:

```
public static ImageIcon[] icons = {new ImageIcon("RedButton.png"),
                                   new ImageIcon("OrangeButton.png"),
                                   new ImageIcon("YellowButton.png"),
                                   new ImageIcon("GreenButton.png"),
                                   new ImageIcon("BlueButton.png"),
                                   new ImageIcon("LightGrayButton.png"),
                                   new ImageIcon("DarkGrayButton.png")};
```

You will want to download the 7 png image files (shown in the above array) from the website when you download this assignment. **BEFORE HANDING IN YOUR CODE ...** make sure that it ALL displays properly on a windows machine (i.e., try it in the tutorial room before handing it in). MAC and Linux computers may not have the same defaults as a Windows pc, and so the default position, sizes and margins of components may differ. You may need to offset the **ImageIcons** or set different margins for your JButtons (check the API). Make sure to hand in the code that works on the windows machine, otherwise you will likely lose marks.

NOTE: Submit all **.java** and **.class** files (and all **.png** images) needed to run. You **MUST NOT use packages** in your code, **nor projects**. Submit ALL of your files in one folder such that they can be opened and compiled individually in JCreator. Some IDEs may create packages and/or projects automatically. You **MUST** export the .java files and remove the package code at the top if it is there. Do NOT submit JCreator projects either. **JUST SUBMIT the JAVA and CLASS FILES**. Note that if your internet connection at home is down or does not work, we will not accept this as a reason for handing in an assignment late ... so make sure to submit the assignment WELL BEFORE it is due !

Please NOTE that you WILL lose marks on this assignment if any of your files are missing. You will also lose marks if your code is not written neatly with proper indentation. See examples in the notes for proper style.