A GUI guide for Blackjack Project

GUI is short for Graphical User Interfaces. User can use GUI to communicate with program. In java, we can use `Swing` or `JavaFX` to build a GUI. In this guide, we use Swing as an example.

1 Swing component

- **JFrame**: Java GUI programs are based on JFrame. It is the object of the window on the screen, which can be maximized, minimized and closed.
- **JPanel**: The panel container class in swing. It is a lightweight Container, can be added to the JFrame form. The function is to combine components with the same logic function in the form.
- **JTextField**: A lightweight component. It allow the user to edit single line text.
- **JButton**: An instance of JButton class. Use to create buttons.

2 Hello world GUI

Code

```java
import javax.swing.JFrame;
import javax.swing.JLabel;
import java.awt.*;

public class HelloWorld extends JFrame{
    public HelloWorld(){
        setTitle("Hello GUI");
        // Set the window size
        setSize(400,200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // Add a label
        JLabel label = new JLabel("Hello world, this is my first swing GUI.");
        Container container = getContentPane();
        container.add(label);
        setVisible(true);
    }

    public static void main(String[] args) {
        new HelloWorld();
    }
}
```

What you got
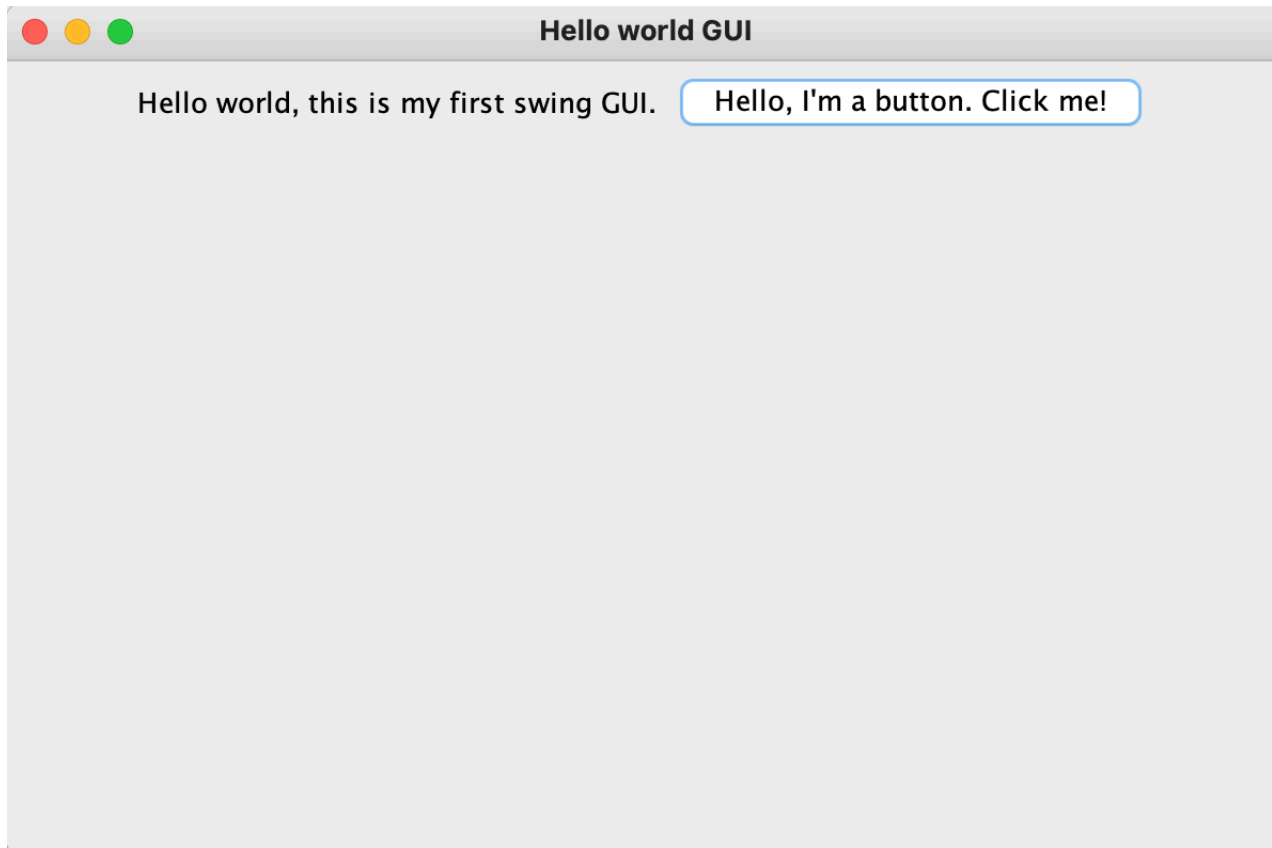
After clicking the button for 5 times,

3 More complex demo

Let's see a more complex demo with image.

Code

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
```

```java
public class ComplexDemo extends JFrame {
    public static void main(String args[]) {
        JFrame frame = new JFrame("Complex demo");
        JPanel panel = new JPanel();
        frame.getContentPane();
        JLabel label = new JLabel();
        // The parameter should be the absolute path of the image file
        ImageIcon img = new ImageIcon("<The absolute path of the image file>");
        img.setImage(img.getImage().getScaledInstance(200,300,
Image.SCALE_DEFAULT));
        // Add the image as the content of the label
        label.setIcon(img);
        label.addMouseListener(new MouseListener() {
            private int click_time = 0;
            @Override
            public void mouseClicked(MouseEvent e) {
                // When mouse click the image
                click_time ++;
                String msg = String.format("You choose Hearts three. You
clicked %d time(s).", click_time);
                JOptionPane.showMessageDialog(null, msg, "Message",
JOptionPane.INFORMATION_MESSAGE);
            }

            @Override
            public void mousePressed(MouseEvent e) {
                // You can write the action when the mouse pressed on the image
                // Leave blank if you don't want to use it
            }

            @Override
            public void mouseReleased(MouseEvent e) {
                // You can write the action when the mouse released from the
image
                // Leave blank if you don't want to use it
            }

            @Override
            public void mouseEntered(MouseEvent e) {
                // You can write the action when the mouse enter the image
                // Leave blank if you don't want to use it
            }

            @Override
            public void mouseExited(MouseEvent e) {
                // You can write the action when the mouse exit from the image
                // Leave blank if you don't want to use it
            }
```
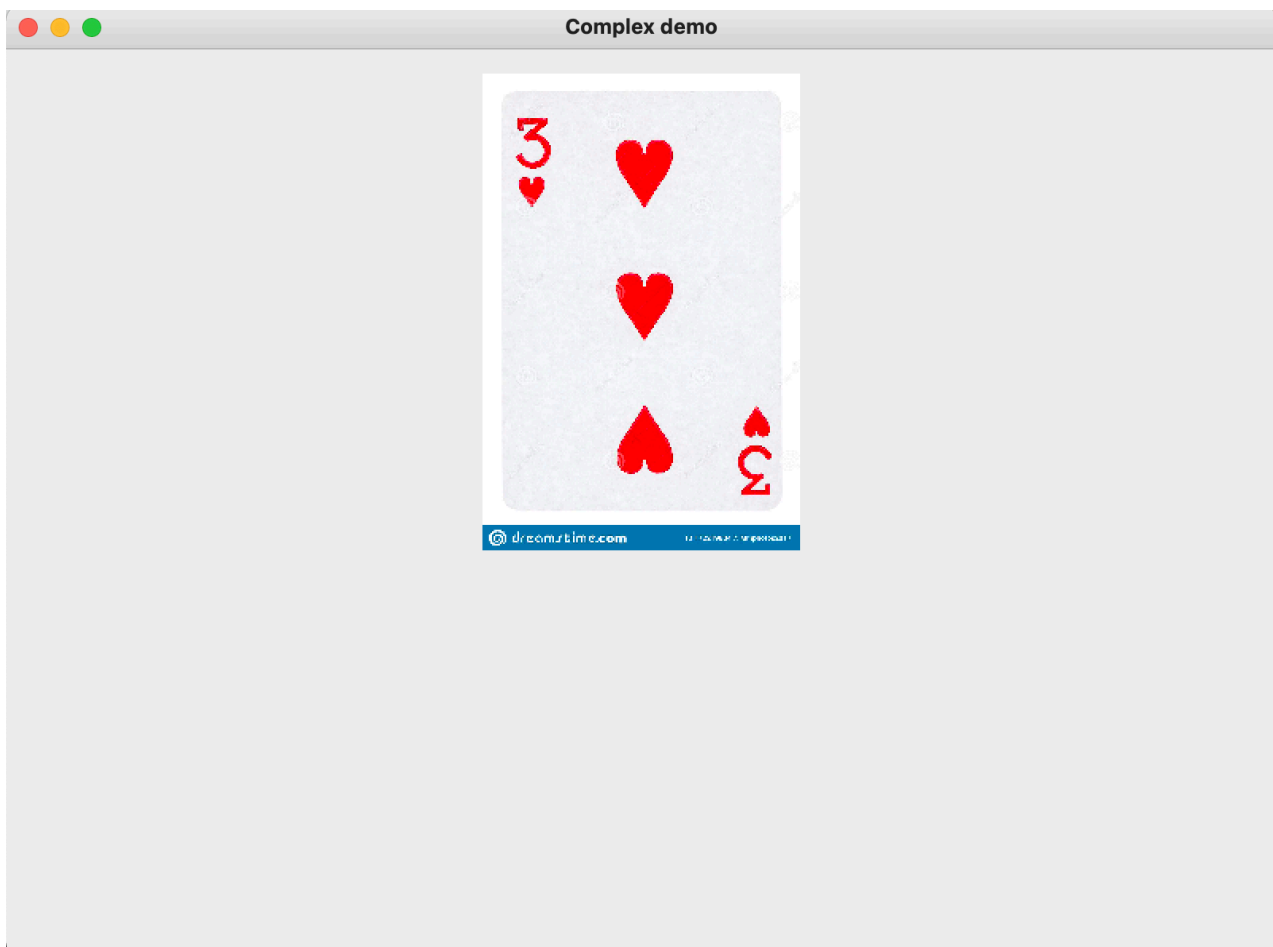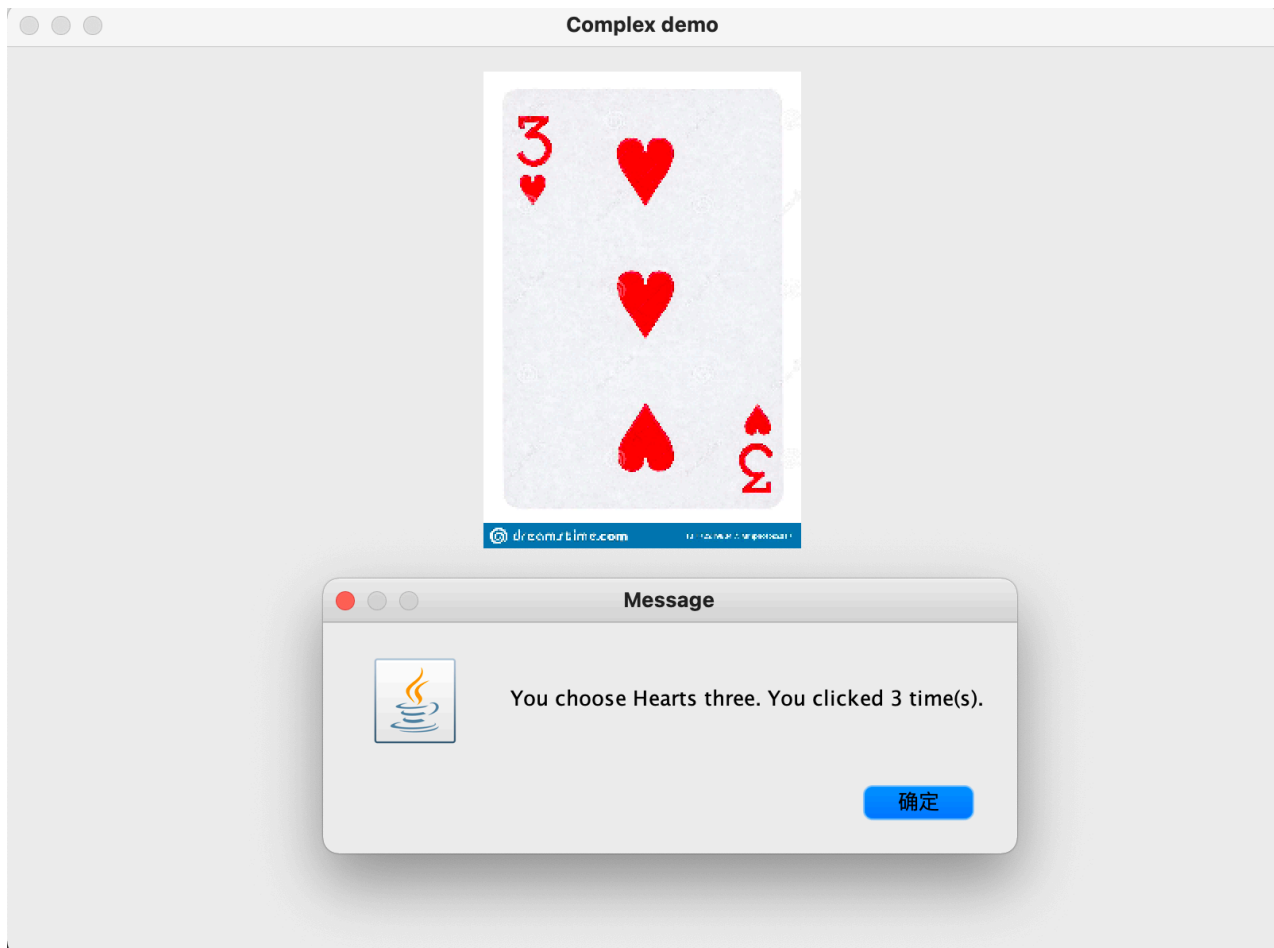
```
        });
        panel.add(label);
        panel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        frame.add(panel);
        frame.setSize(800, 600);
        frame.setVisible(true);
    }
}
```

What you got

After clicking the image for three times,

Good luck and have fun!