Advanced Programming Term Project Report

My JPainting

Oğuzhan Oktay BÜYÜK

1221221018

A report submitted in part fulfilment of the term project of

Advanced Programming in

Computer Engineering

Instructor Name: Prof. Dr. Akif EYLER

Department of Computer Engineering

University of Fatih Sultan Mehmet V.

December 23, 2015

Declaration

This report has been prepared on the basis of my own work. I used mostly Java Graphics concept that was learnt in Advanced Programming Lesson.

Word Count[Total]: 1360

Student Name: Oğuzhan Oktay Büyük

Date of Submission: 23.12.2015

Table of Contents

Abstract

The purpose of the this paper to clearly explain the improvement on term project of Advanced Programming Lesson using Java Frameworks. The project is based on painting to understand the concept of Java Graphics. The project called as *My JPainting* is a Java based project. Java is learnt as *highly level* in Advanced Programming Lesson. In a part of the lesson, Graphics and Mouse Listener events were taught so my decision on the project is based on how these things can be used. The project is based on Graphics, Mouse Listener and File usage(create, open, save etc.). The project is mainly like painting program moreover it has a good graphical user interface and introduction conversation[wav file].

This project is useful in terms of quickly painting, correction of panel background color. The program includes some kind of things that enables to paint just using mouse actions, listeners. The Project provides quickly drawable shapes like square, rectangle. Also, the program is small sized just click and run it.

The project mainly focused on Graphics usage and mouse actions. Files' operations are used in program that saving your painting for further usage. By using Java, small sized and quickly painting program can be used for clearly understanding of Graphics concept.

Keywords: Graphics, MouseListener, File Operations on Java, Java Sound API,

Project Specification

The project specification, in which what the project uses, is defined as the following steps:

- SSS(Small Simple Safe by Mr. Akif Eyler)
- NetBeans IDE is used for development stage.
- Java GUI
- Java Graphics
- Java Mouse Actions
- Java File Operations
- Java Sound API

SSS is the lesson software resource. It is used for understanding better the concept of Java and Java Framework. It has simple usage and good graphical interface that a programmer can understand it well. At the point of Java Framework, I have searched the class hierarchy from this software that I understood and learnt where some imports [ex. Import java.awt.event.ActionListener, java.io.BufferedWriter] come from.

NetBeans IDE is the program that the project was implemented and built in. It is also understandable and convenient user interface for the users.

Java GUI is the Swing. On swing, it has benefits as well as some restrictions but benefits of it is hugely efficient.

Java Graphics class is very important class that allows an application to draw onto components that are made realized on some kind of devices. Java 2D API is very powerful as well as complex. Graphics class has important two groups such as:

- Draw and fill methods, enabling you to render basic shapes, text, and images
- Attributes setting methods, which affect how that drawing and filling appears

Java Mouse Actions can play an impact role while controlling of your GUI. Java Mouse Listener is the mainly used in the project. Java Mouse events notify when the user uses the mouse to interact with a component just like frame, panel, shape, or something in area of component.

Java File Operations are critical on the application when you want to save your painting and if you want to continue later, you need to store it in a file so the project uses Buffered Writer, InputStream, OutputStream concepts.

Java Sound API is used for introduction of the project and also to listen music which has .wav extension.

Chapter 1: **Introduction**

The project is a painting program on Java platform. The program mainly provides with drawing your imagination on a panel. The program includes colourful areas to identify which color you want. All things are dynamically changed by just clicking mouse. Mouse Listener plays an big role on drawing something like getX() and getY() of mouse movement. It has features like drawing with a pencil, drawing with a brush, erasing graphics, painting the all area, getting color of the pane or the drawing items, draw shapes such as square, rectangle, ellipse and line. You can adjust the stroke of the paintings just using JSlider component.

To use the drawing items, we need to create ***draw*** method for drawing line ***drawLine(),*** drawing rectangle with ***drawRect()*** method. To understand the concept of Graphics and dynamic update, we need to use ***paintComponent()***, ***update()*** the graphics. Update() method gets ***Graphics g*** as a parameter. In this method, we need to define the update configurations on drawing items.

The application use JColorChooser that enables to select which color we want. This thing is set up to using shapes on updating. Therefore, it is colored as selected.

In the application, I use threads while drawing panel with implementing ***Runnable interface***. Moreover, threads are used while music file is running as well as doing another thing on panel.

MouseEvents are really importing because every step of the program consisting of mouse listener events. For example, when you mouse pressed to panel it controls with boolean variable that it is changed or not. After that step, it starts startingThread method to continuously doing the selected thing.

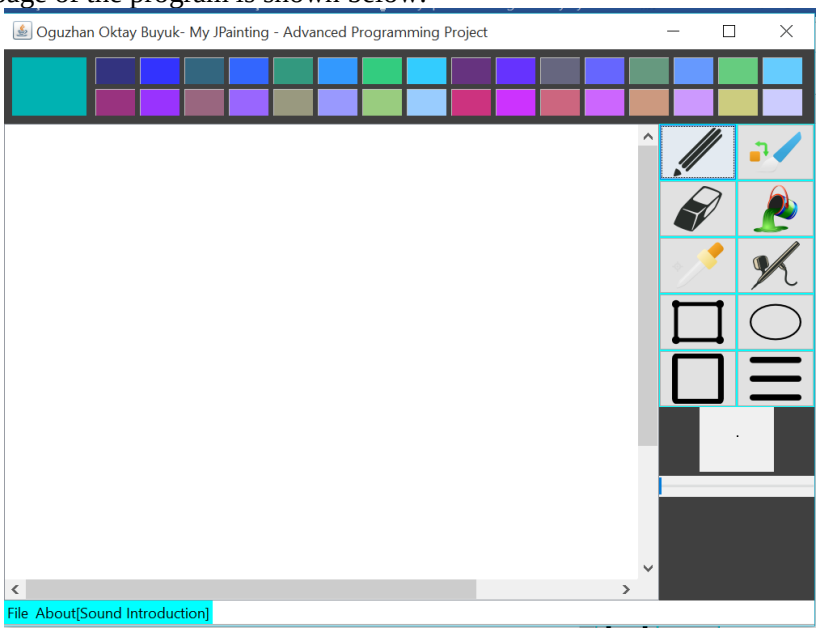
Chapter 2: Usage of Application

To begin the introduction application with visual images as following:

1. The project has a ***Splash Screen*** that waits the loading of the Main application.

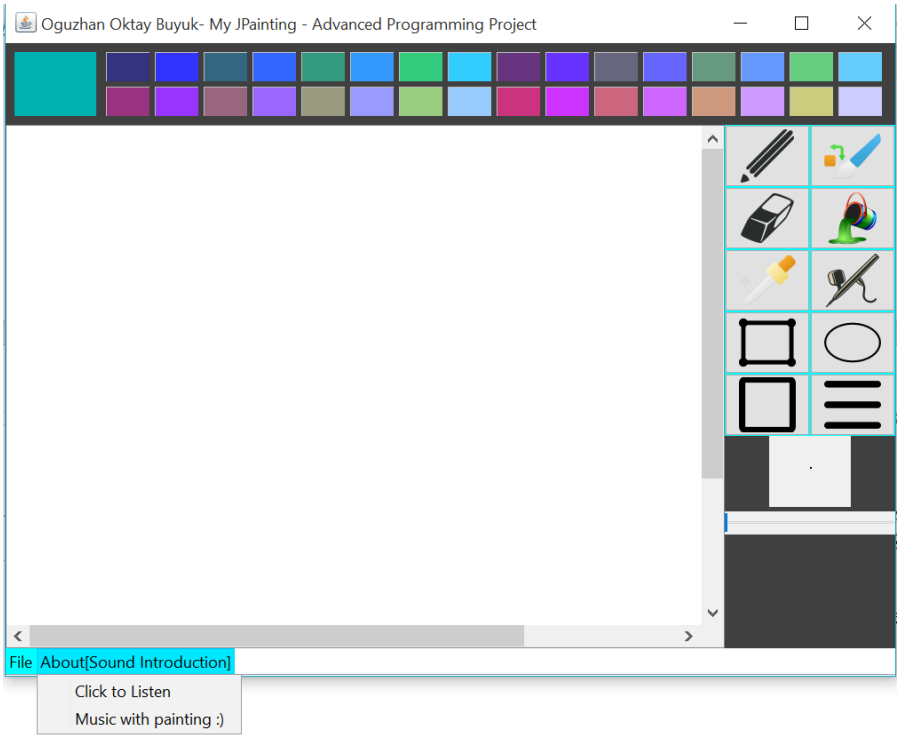
- Waiting is realized using ***Thread.sleep()*** method.

2. The main page of the program is shown below:



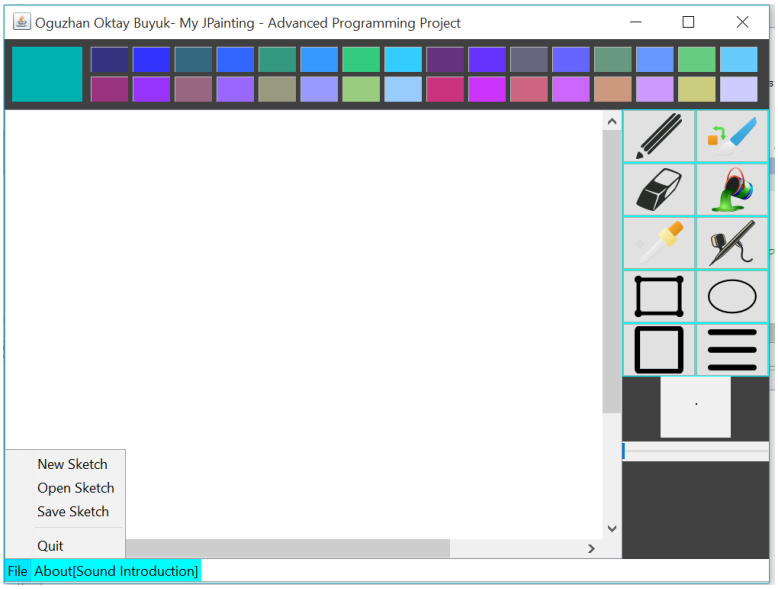
As you can see main page, the program has a Panel, Labels for colors and tools. There is a JSlider for sizing the stroke of shapes. There is a Label leftmost which is JColorChooser to identify the color that user specified. There is also a JMenu.

3. Getting introduction ***wav(Java Sound API supports .wav file not mp3)*** music is provided by clicking About[Sound Introduction]



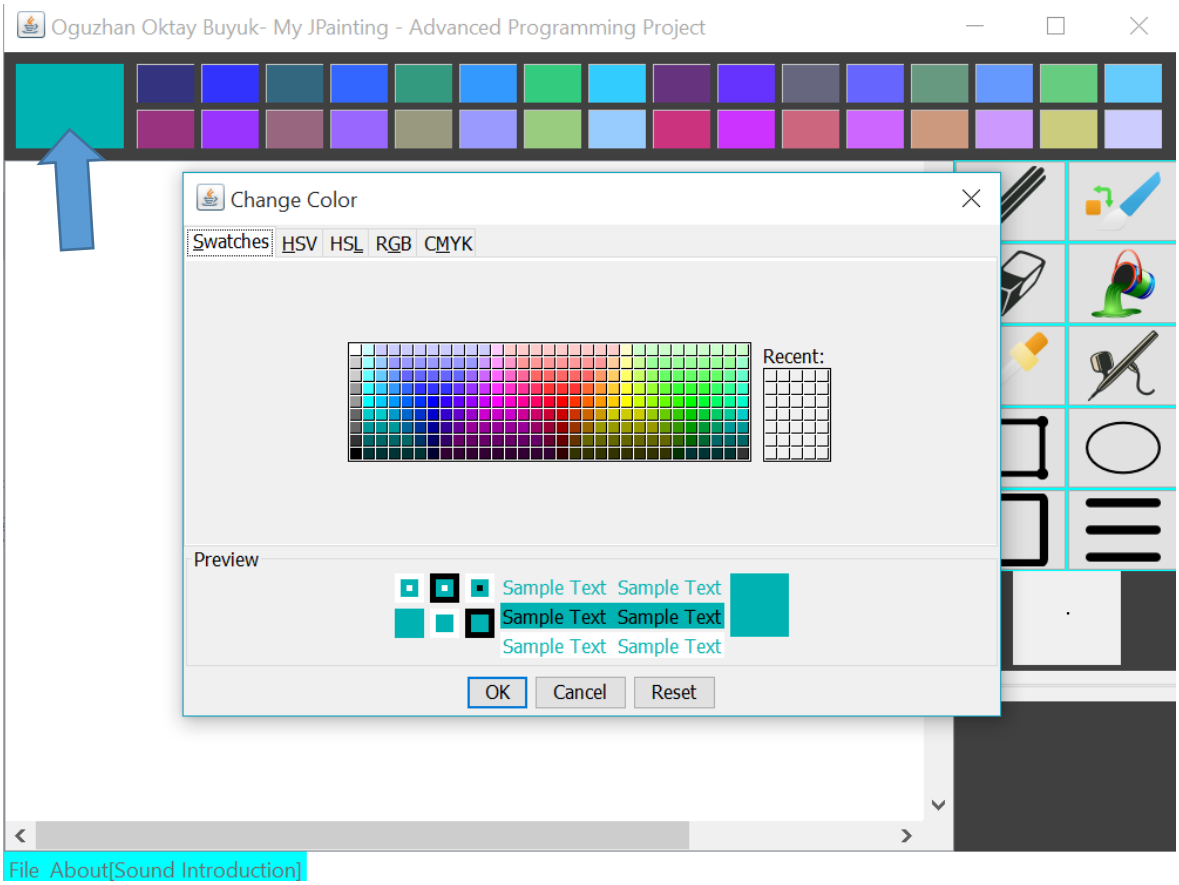
By clicking *Click to Listen*, you will come across a JFileChooser to choose file ***introProject.wav*** that added to Github folder(you can download it.) Music with painting JMenuItem opens JFileChooser to open music.wav file that you can play any music extension with .wav.

4. File operations controlled by File menu.

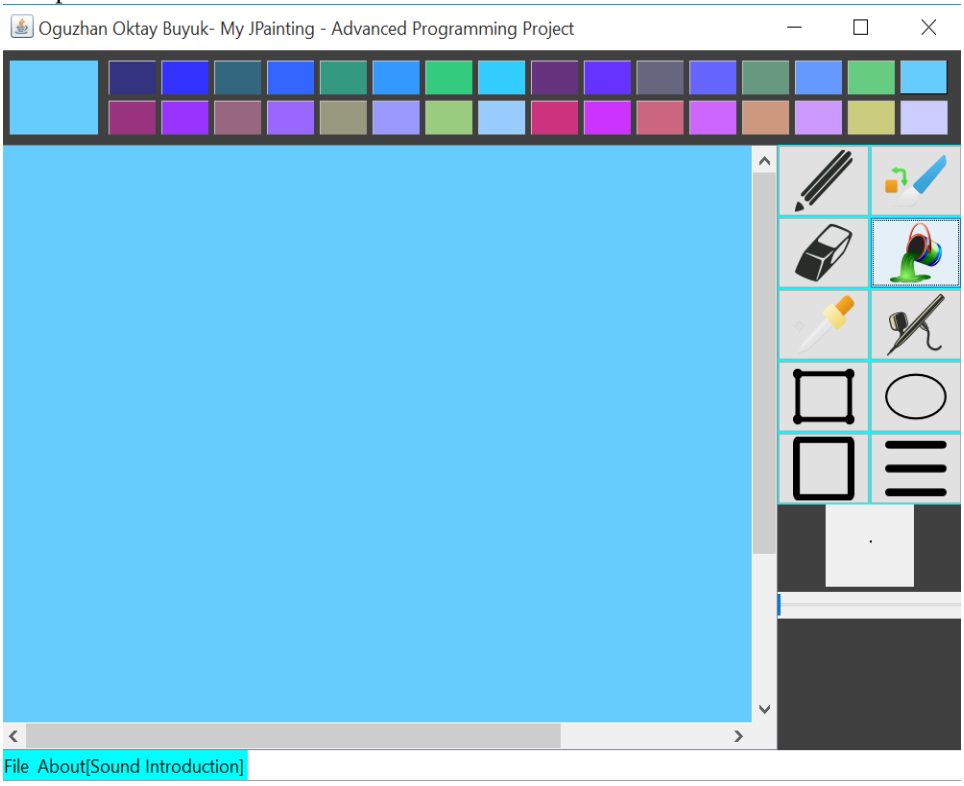


New Sketch clears all panel. *Open Sketch* provides with opening saved sketch from before saved items. *Save Sketch* is to save for further corrections. *Quit* is to quit from program.

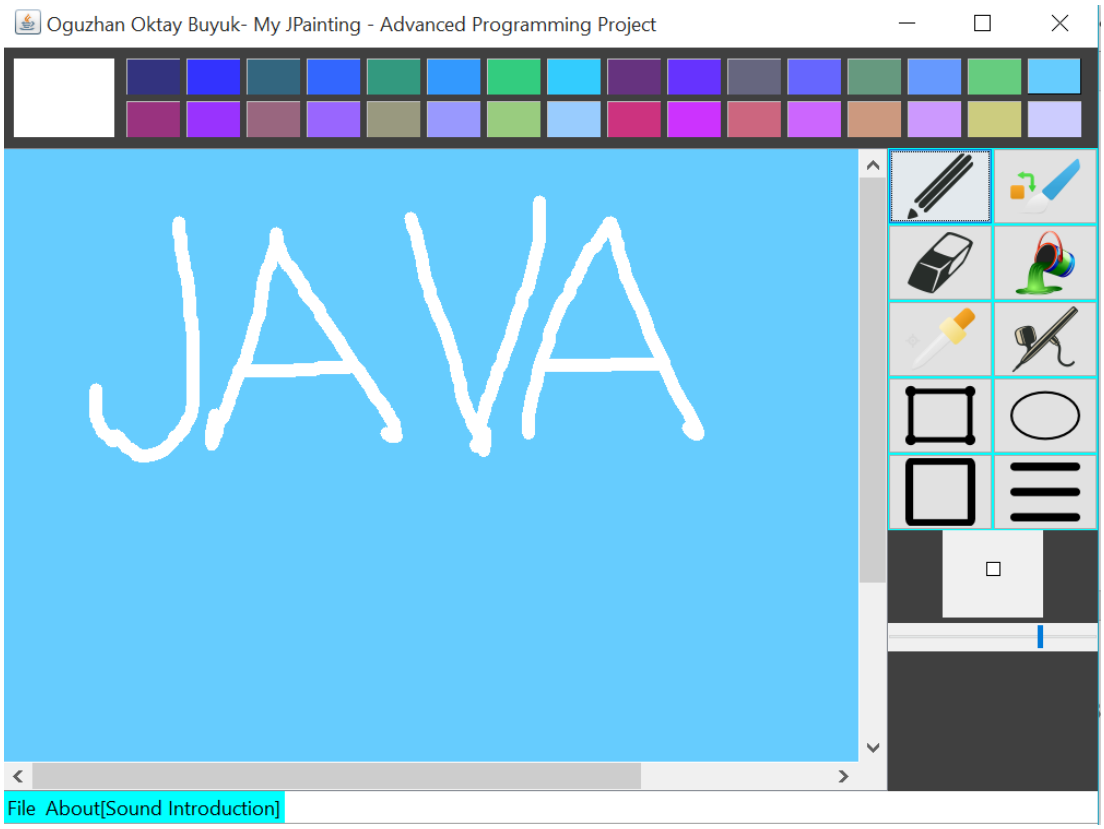
5. ColorChooser from clicking leftmost Label.



6. For example we selected blue color from ColorChooser and clicked the bucket  .



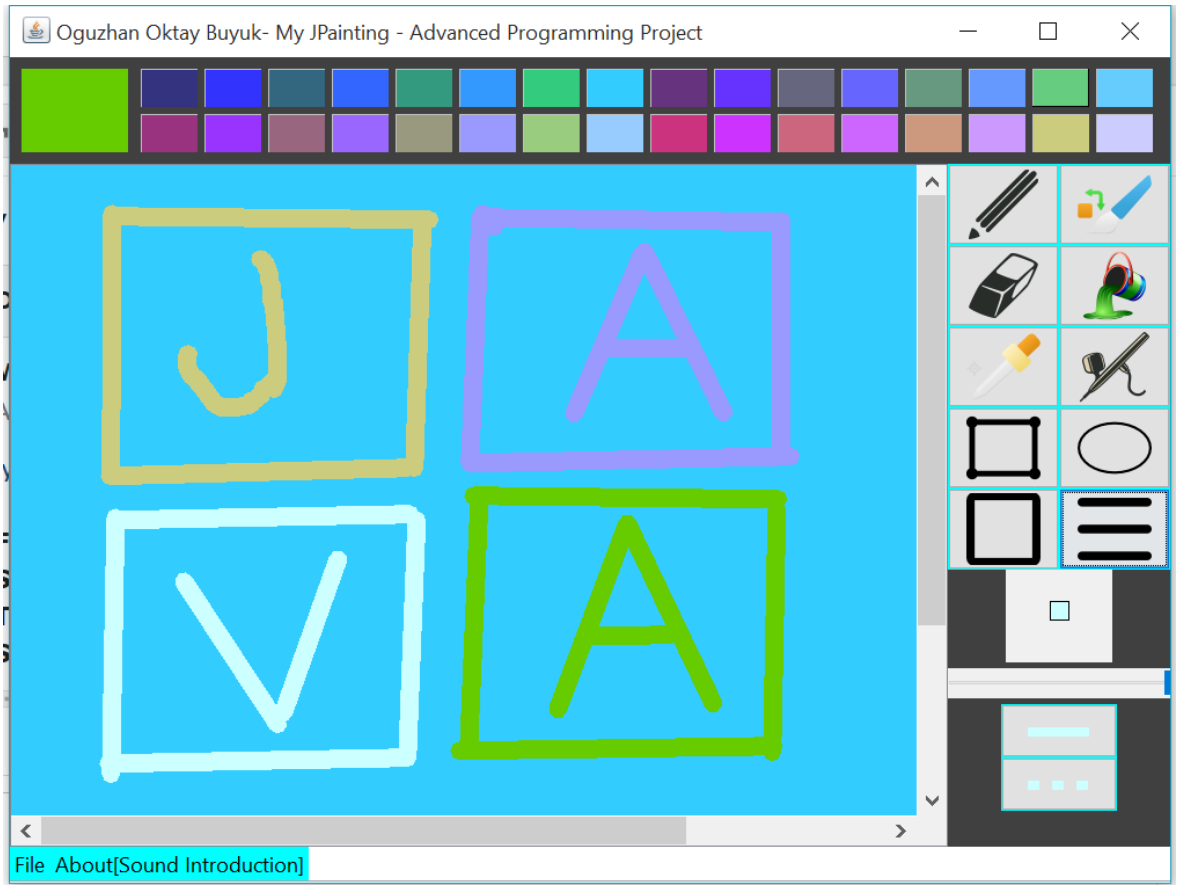
7. We can draw something using pencil.  For example we can write JAVA with this pencil.



8. With brushing tool, you can paint like that:

This was written clicking brushing tool.

9. Using shapes to draw like that:



2.1 Source Code

Some code pieces that identify Graphics usage are here:[The source code are separated from each other. These are the sample example of how I used the methods.]

//Identify of paint Component

```
  public void paintComponent(Graphics g) {
        super.paintComponent(g);
    }
```

```
//for oval painting.
g.fillOval(getWidth() / 2, getHeight() / 2, getWidth() / 2, getHeight() /
2 * 3);

//for Rectangle painting.
g.fillRect(getWidth() / 2 * 3, getHeight() / 2 getWidth() / 2, getHeight()
/ 2* 3);

//for Rectangle drawing.
g.drawRect(getWidth() / 3, getHeight() / 3, getWidth() / 3 * 2, getHeight()
/3 * 2);
```

// thread is starting.

```
  public void threadFetch() {
```

```
        this.playBack = new Thread(this);
        this.playBack.start();
    }
```

//For sounding operations

```
Line.Info info = new Line.Info(Clip.class);
Line strL = AudioSystem.getLine(info);
Clip playM = (Clip) strL;
AudioInputStream audio = AudioSystem.getAudioInputStream(soundFile);
playM.open(audio);
playM.start();
```

Reference

[1] Knudsen, J. (1999) Chapter 9. Images, *Java 2D Graphics. p.* 153–155.

[2] Trail: 2D Graphics, Available: https://docs.oracle.com/javase/tutorial/2d/
[Accessed 11 December 2015]

[3] Class Graphics, Available: https://docs.oracle.com/javase/7/docs/api/java/awt/Graphics.html
[Accessed 11 December 2015]

[4] Java Sound API, Available: https://docs.oracle.com/javase/tutorial/sound/
[Accessed 14 December 2015]

[5] The Java Tutorials, *How to Write a Mouse-Motion Listener*,
Available: https://docs.oracle.com/javase/tutorial/uiswing/events/mousemotionlistener.html
[Accessed 15 December 2015]