**Sprint #0 Report**

Instructions

**Objectives**

* Make decisions on the SOS software development project.
* Learn unit testing and GUI programming in the language of your choice.

**Deliverables and Grading Policy**

Read the “CS 449 Homework Overview” document **carefully** and make the key decisions for the software development. Use the following template to complete your report.

1. **Key Decisions of the SOS Project (2 points)**

| Object-oriented programming language | Java |
| --- | --- |
| GUI library (strongly encouraged) | Swing |
| IDE (Integrated Development Environment) | Eclipse |
| xUnit framework (e.g., JUnit for Java) | JUnit |
| Programming style guide (must read it carefully) | Google Java Style |
| Project hosting site | Github.com |
| Other decisions if applicable |  |

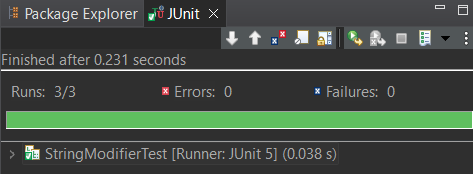
Sample programming style guides:

* Google Java Style Guide: <https://google.github.io/styleguide/javaguide.html>
* Google C++ Style Guide: <https://google.github.io/styleguide/cppguide.html>
* Google Python Style Guide: <https://google.github.io/styleguide/pyguide.html>

1. **Unit testing (4 points)**

Find a tutorial on the unit test framework you have chosen and write at least two xUnit tests of a program you have written or found elsewhere. Attach here (1) the screenshot of your program execution and (2) the source code of your program.

**Screenshot:**



**StringModifier.java (class to be tested):**

package stringmodifier;

public class StringModifier {

/\*\*

\* Returns a new string consisting of the original string concatenated onto itself a given number of times.

\* @param string String to be repeated

\* @param repeatNumber Number of times the given string will be repeated

\* @return Self-concatenated string

\*/

public String unmodifiedString(String string) {

return string;

}

public String repeatedString(String string, int repeatNumber) {

String newString = "";

for (int i = 0; i < repeatNumber; i++) {

newString += string;

}

return newString;

}

}

**StringModifierTest.java (JUnit test class):**

package stringmodifier;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

class StringModifierTest {

StringModifier stringModTest = new StringModifier();

@Test

void testUnmodifiedString() {

assertEquals("Johnathan hello", stringModTest.unmodifiedString("Johnathan hello"));

}

@Test

void testRepeatedString() {

assertEquals("JohnJohnJohn", stringModTest.repeatedString("John", 3));

}

@Test

void testRepeatedStringEmpty() {

assertEquals("", stringModTest.repeatedString("John", 0));

}

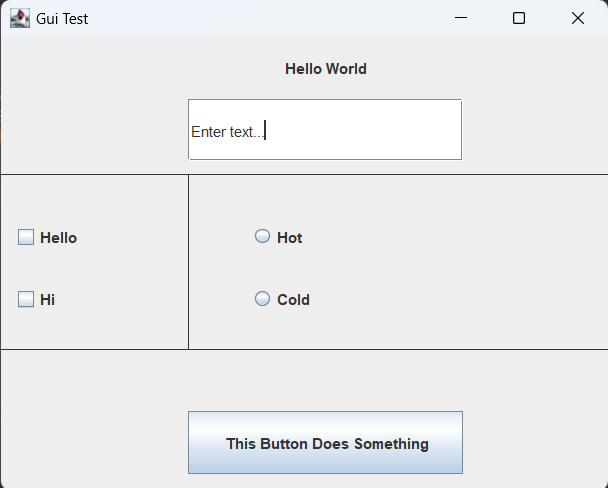
}

1. **GUI programming (4 points)**

Write a GUI program in the language you have chosen for your SOS project. The GUI of your program must include text, lines, a check box, and radio buttons. While you are recommended to consider the GUI for the SOS game board, it is not required. In this assignment, any GUI program of your own work is acceptable.

Attach here (1) the screenshot of your program execution and (2) the source code of your program.

**Screenshot:**



**Gui.java:**

package sprint0;

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Gui {

{

// Creating instance of JFrame

JFrame frame = new JFrame("Gui Test");

// Label with text

JLabel label1 = new JLabel("Hello World");

label1.setBounds(150, 0, 220, 50);

label1.setHorizontalAlignment(SwingConstants.CENTER);;

frame.add(label1);

// Text field

JTextField textField1 = new JTextField("Enter text...");

textField1.setBounds(150, 50, 220, 50);

frame.add(textField1);

// Check boxes

JCheckBox checkBox1 = new JCheckBox("Hello");

JCheckBox checkBox2 = new JCheckBox("Hi");

checkBox1.setBounds(10, 150, 100, 20);

checkBox2.setBounds(10, 200, 100, 20);

frame.add(checkBox1);

frame.add(checkBox2);

// Radio buttons

JRadioButton radio1 = new JRadioButton("Hot");

JRadioButton radio2 = new JRadioButton("Cold");

radio1.setBounds(200, 150, 100, 20);

radio2.setBounds(200, 200, 100, 20);

ButtonGroup radioGroup = new ButtonGroup();

radioGroup.add(radio1);

radioGroup.add(radio2);

frame.add(radio1);

frame.add(radio2);

// Button that changes label text

JButton button1 = new JButton("This Button Does Something");

button1.setBounds(150, 300, 220, 50);

frame.add(button1);

button1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

label1.setText("You pressed button!!!");

}

});

frame.setSize(500, 400);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Draw lines

frame.getContentPane().add(new Lines ());

frame.setVisible(true);

}

// Draw dividing lines

@SuppressWarnings("serial")

class Lines extends JPanel {

@Override

public void paintComponent(Graphics g) {

super.paintComponent(g);

g.drawLine(0, 110, 500, 110);

g.drawLine(0, 250, 500, 250);

g.drawLine(150, 110, 150, 250);

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

public void run() {

new Gui();

}

});

}

}