Unit Testing using JUNIT

BODY MASS INDEX CALCULATOR

—

Presented by:

Mohammed Afnan Althaf Samad - 200911006
Anushka Gupta - 200911070

OVERVIEW

This report presents unit testing for functionalities of our chosen application, Body Mass Calculator, a simple JAVA Swing application. The unit testing has been done on eclipse using the JUNIT Software testing tool.

Body mass index (BMI) is a measure of body fat based on height and weight that applies to adult men and women. The BMI is defined as the body mass divided by the square of the body height, and is expressed in units of kg/m2, resulting from mass in kilograms and height in metres.

SOURCE CODE

```
package com.company;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.text.*;

public class Main extends JFrame {
    Container c;
    JLabel lbKG, lblFt, lblln, lblHT;
    JTextField txtKg;
    JComboBox cbFt, cbln;
    JPanel p1, p2, p3;
    JButton btnSubmit;
    Main() {
        c = getContentPane();
        c.setLayout(new BoxLayout(c, BoxLayout.Y_AXIS));
        p1 = new JPanel(new FlowLayout(FlowLayout.LEFT));
        lbKG = new JLabel("" +
                "Weight in Kg:");
        txtKg = new JTextField(10
        );
        p1.add(lbKG);
        p1.add(txtKg);
        c.add(p1);
        p2 = new JPanel(new FlowLayout(FlowLayout.LEFT));
        String[] ft = {"1", "2", "3", "4", "5", "6", "7"};
        String[] in = {"0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11"};
        lblHT = new JLabel("Height");
```

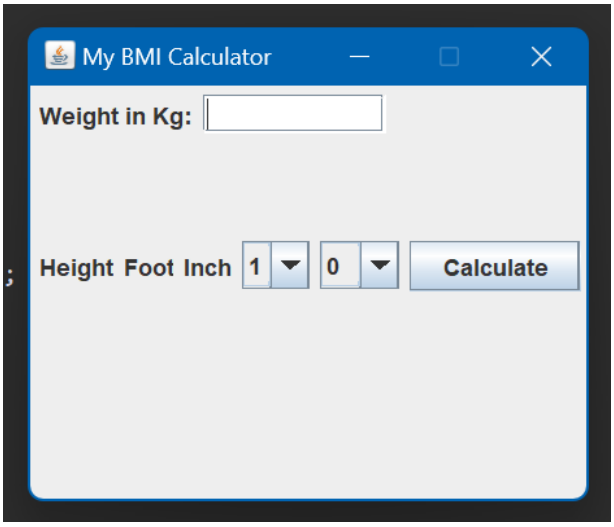
```
        cbFt = new JComboBox(ft);
        cbIn = new JComboBox(in);
        lblFt = new JLabel("Foot");
        lblIn = new JLabel("Inch");
        p2.add(lblHT);
        p2.add(lblFt);
        p2.add(lblIn);
        p2.add(cbFt);
        p2.add(cbIn);
        c.add(p2);
        p3 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        btnSubmit = new JButton("Calculate");
        p2.add(btnSubmit);
        c.add(p3);
        btnSubmit.addActionListener(new L1());
        p3 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        btnSubmit = new JButton("Close");
        p2.add(btnSubmit);
    }
class L1 implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (txtKg.getText().equals("")) {
            JOptionPane.showMessageDialog(c, "Weight should not be blank");
            txtKg.requestFocus();
        } else {
            try {
                double weight = Double.parseDouble(txtKg.getText());
                if (weight <= 0) {
```

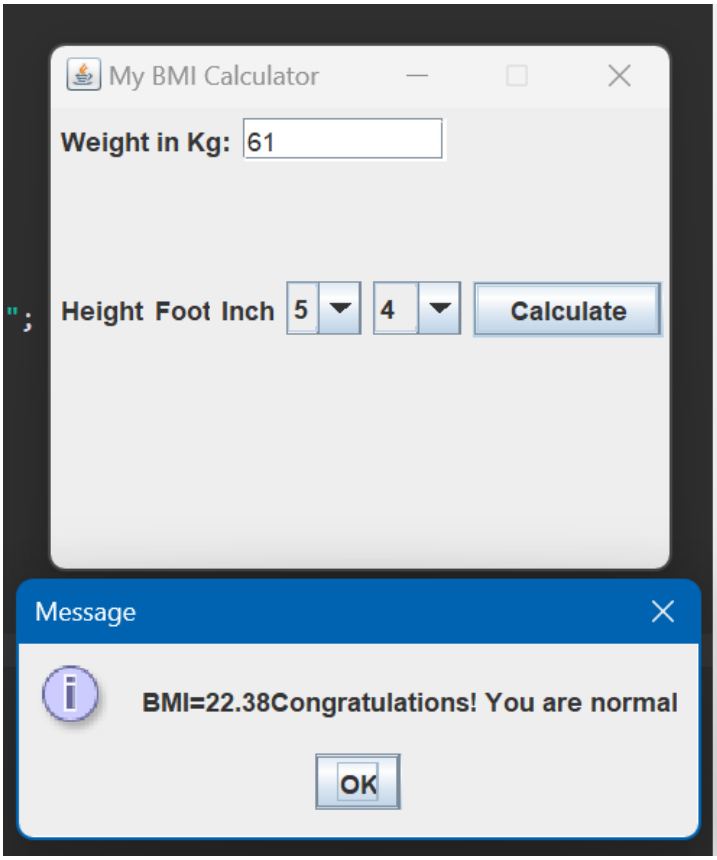
```
            JOptionPane.showMessageDialog(c, "Weight should be greater than 0");
            txtKg.setText("");
            txtKg.requestFocus();
        }
        else {
            Object ftitem = cbFt.getSelectedItem();
            String ft = (String) ftitem;
            int foot = Integer.parseInt(ft);
            Object lnitem = cbln.getSelectedItem();
            String ln = (String) ftitem;
            int Inches = Integer.parseInt(ln);
            //convert FT to inches
            double height=findheight(foot,Inches);
            double bmi =findbmi(weight,height);
            String msg=findmsg(bmi);
            NumberFormat nf = NumberFormat.getInstance();
            nf.setMaximumFractionDigits(2);
            String bmis = nf.format(bmi);
            JOptionPane.showMessageDialog(c, "BMI= " + bmis + " . " + msg);
            txtKg.setText("");
            cbFt.setSelectedItem("1");
            cbln.setSelectedItem("0");
        }
    }
    catch(NumberFormatException error){
        JOptionPane.showMessageDialog(c,"Enter proper weight");
        txtKg.setText("");
        txtKg.requestFocus();;
    }
```

```
    }
  }
}
static double findheight(int foots, int inches)
{
        while (foots > 0) {
    inches += 12;
    foots--;
  }
        double height=inches*2.54;
        return height;
}
static double findbmi(double weights, double heights)
{
        double bmi=weights / (heights * heights);
        bmi = bmi * 10000;
        return bmi;
}
static String findmsg(double bmis)
{
        String msg;
        if (bmis < 18.5) msg = "You are underweight";
  else if (bmis >= 18.5 & bmis < 25) msg = "Congratulations! You are normal";
  else if (bmis >= 25 && bmis < 30) msg = "You are overweight";
  else msg = "You are Obese";
        return msg;
}
  public static void main(String[] args) {
    Main b= new Main();
```

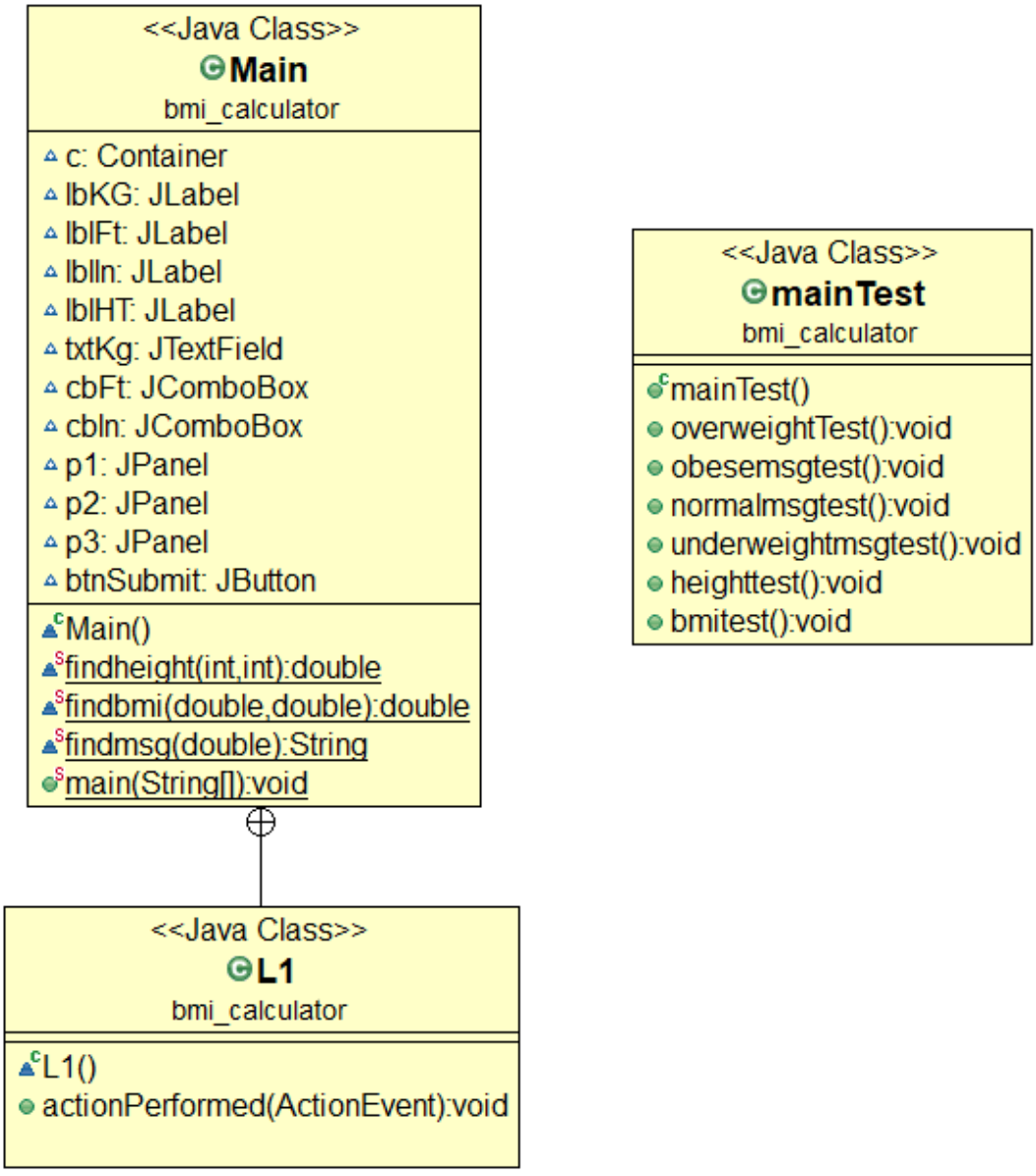
```
        b.setSize(300,250);
        b.setVisible(true);
        b.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        b.setTitle("My BMI Calculator");
        b.setLocation(300,300);
        b.setResizable(false);
    }
}
```

UI SCREENSHOTS





CLASS DIAGRAM



TEST CASES FOR FUNCTIONALITIES

We have considered 6 test cases for the BMI calculator. They are to test overweight, obese, normal, underweight, height and BMI conditions.

overweightmsgtest() - runs without failure if (bmis>=25 && bmis<30)

obesemsgtest() - runs without failure if (bmis>=30)

normalmsgtest() - runs without failure if (bmis>=18.5 && bmis<25)

underweightmsgtest() - runs without failure if (bmis<18.5)

TEST SCRIPT

```
package bmi_calculator;
import static org.junit.Assert.*;
import org.junit.Test;
import org.junit.Test;

public class mainTest {
	@Test
	public void overweightTest() {
		var msg=new Main();
		assertEquals("You are overweight",msg.findmsg(25));
	}
	@Test
	public void obesemsgtest() {
		var msg=new Main();
		assertEquals("You are Obese",msg.findmsg(30));
	}
	@Test
	public void normalmsgtest() {
		var msg=new Main();
		assertEquals("Congratulations! You are normal",msg.findmsg(18.5));
	}
	@Test
	public void underweightmsgtest() {
		var msg=new Main();
		assertEquals("You are underweight",msg.findmsg(0));
	}
```

```
	//tests for find height function
	@Test
	public void heighttest() {
		var height=new Main();
		assertEquals(71.12,height.findheight(2,4),0.0);
	}
	//tests for find bmi function
	@Test
	public void bmitest() {
		var bmi=new Main();
		assertEquals(10000,bmi.findbmi(25,5),0);
	}
}
```

TEST REPORT FROM JUNIT (USING MAVEN)

CASE -1 (NO FAILURES)

T E S T S

Running bmi_calculator.mainTest

Tests run: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.004 sec

Results :

Tests run: 6, Failures: 0, Errors: 0, Skipped: 0

[INFO] --

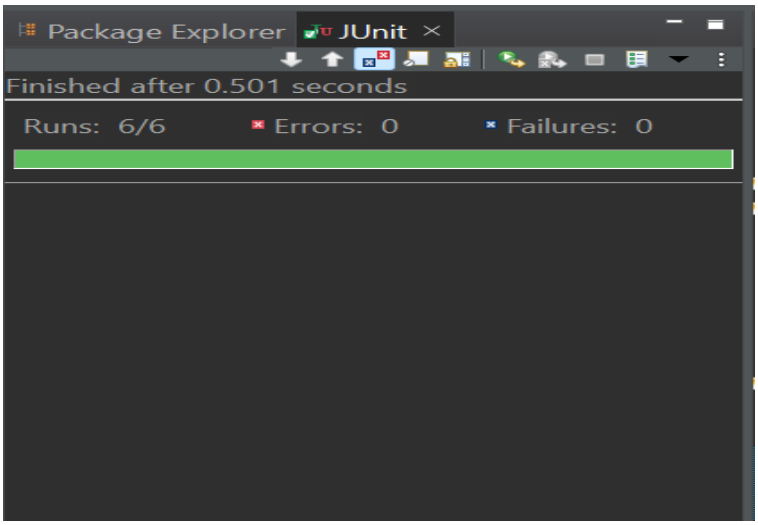
[INFO] BUILD SUCCESS

[INFO] --

[INFO] Total time: 2.102 s

[INFO] Finished at: 2022-11-07T17:52:14+05:30

[INFO] --



CASE -2 (WITH FAILURES)

T E S T S

Running bmi_calculator.mainTest

Tests run: 6, Failures: 2, Errors: 0, Skipped: 0, Time elapsed: 1.113 sec

Results :

Tests run: 6, Failures: 2, Errors: 0, Skipped: 0

[INFO] --

[INFO] BUILD SUCCESS

[INFO] --

[INFO] Total time: 2.102 s

[INFO] Finished at: 2022-11-07T17:58:25+05:30

[INFO] --