Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический университет

“ЛЭТИ” им.В.И.Ульянова (Ленина) »

Кафедра ВТ

**ОТЧЕТ**

**по лабораторно-практической работе № 9**

**«Модульное тестирование приложения»**

**по дисциплине «Объектно - ориентированное программирование на языке Java»**

Выполнила Половникова А.С.

Факультет КТИ

Группа № 3312

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2024 г

**Цель работы**

Знакомство с технологией модульного тестирования Java-приложения с использование системы JUnit.

**Описание задания**

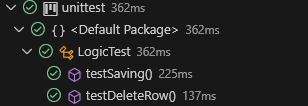
1. Перечень методов, которые тестируются в приложении.
2. Скриншоты, иллюстрирующие выполнение тестов.
3. Текст документации, сгенерированный в Javadoc.
4. Исходный код класса тестов и тестирующих методов.

**Описание приложения**

В приложение будут тестироваться методы saving и b2Click класса logic. Они будут тестироваться методами testSaving и testDeleteRow класса LogicTest. Тестирующий метод testSaving будет проверять в методе saving успешность сохранения в таблице. Метод testDeleteRow буте проверять в методе b2Click успешность удаления первой строки*.*

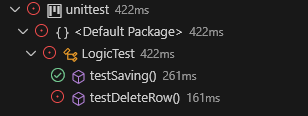
**Описание проверки работоспособности приложения**

Запустим JUnit тест и получим следующие

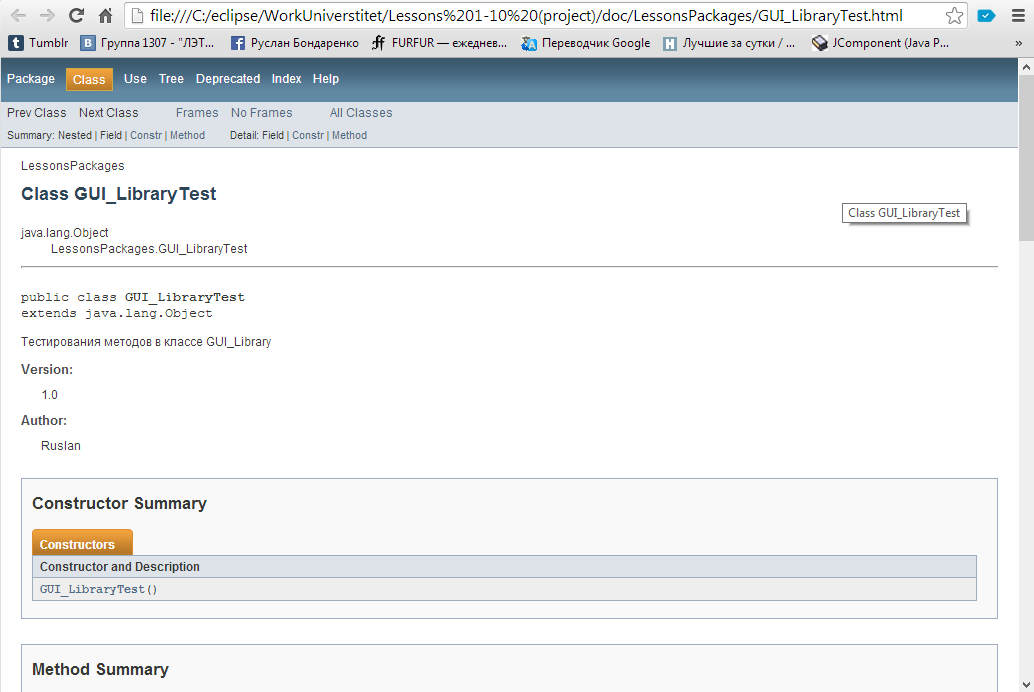


Все тесты выполнены правильно.

Теперь изменим в методе testDeleteRow значение удаляемой строки, в результате получим ошибку. JUnit указывает, где произошла ошибка и почему.



**Текст документации Javadoc**

****

**Исходный код класса тестов и тестирующих методов**

**Код класса тестов**

import static org.junit.Assert.\*;

import javax.swing.JFrame;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;

import org.junit.Before;

import org.junit.Test;

import com.proghelp9.Main;

import com.proghelp9.logic;

/\*\*

 \* @author Половникова Алиса 3312

 \* @version 2.0

 \*/

public class LogicTest

{

    /\*\*

     \* Подготовка окружения перед выполнением тестов.

     \* Инициализирует модель таблицы, массив данных и счетчик строк.

     \*/

    @Before

    public void setUp()

    {

        Main.quant = 0;

        Main.data = new String[0][3];

        Main.model = new DefaultTableModel(new Object[][]{}, new String[]{"Пациент", "Время", "Врач"});

        Main.table = new JTable(Main.model);

    }

    /\*\*

     \* Проверяет добавление новой строки в таблицу.

     \* <p>

     \* Ожидается, что количество строк увеличится на 1, а значения новой строки

     \* будут соответствовать переданным параметрам.

     \*/

    @Test

    public void testSaving()

    {

        logic.addWindow = new JFrame();

        logic.saving(new MockTextField("Иванов"), new MockTextField("10:00"), new MockTextField("Петров"));

        assertEquals("Количество строк должно быть 1", 1, Main.model.getRowCount());

        assertEquals("Иванов", Main.model.getValueAt(0, 0));

        assertEquals("10:00", Main.model.getValueAt(0, 1));

        assertEquals("Петров", Main.model.getValueAt(0, 2));

    }

    /\*\*

     \* Проверяет удаление выбранной строки из таблицы.

     \* <p>

     \* Ожидается, что после удаления останется 1 строка, а ее значения будут корректны.

     \*/

    @Test

    public void testDeleteRow()

    {

        Main.model.addRow(new Object[]{"Иванов", "10:00", "Петров"});

        Main.model.addRow(new Object[]{"Сидоров", "11:00", "Иванов"});

        Main.table.setRowSelectionInterval(0, 0);

        logic.b2Click();

        assertEquals("Количество строк должно быть 1", 1, Main.model.getRowCount());

        assertEquals("Сидоров", Main.model.getValueAt(0, 0));

        assertEquals("11:00", Main.model.getValueAt(0, 1));

        assertEquals("Иванов", Main.model.getValueAt(0, 2));

    }

    /\*\*

     \* Вспомогательный класс для тестирования методов, использующих {@link javax.swing.JTextField}.

     \* Позволяет передать заранее определенное значение текста.

     \*/

    private static class MockTextField extends javax.swing.JTextField

    {

        private final String text;

        /\*\*

         \* Конструктор, принимающий текстовое значение.

         \*

         \* @param text текст, который будет возвращен методом {@link #getText()}.

         \*/

        public MockTextField(String text)

        {

            this.text = text;

        }

        /\*\*

         \* Возвращает заранее установленное значение текста.

         \*

         \* @return текстовое значение.

         \*/

        @Override

        public String getText()

        {

            return text;

        }

    }

}

**Вывод**

Во время выполнения данной лабораторной работы, были получены навыки в технологии модульного тестирования Java-приложения с использование системы JUnit. В пункте «Описание проверки работоспособности приложения» ошибок выявлено не было.