Пример 3: Программа "Assertions Demo"

Цель примера

Продемонстрировать базовый принцип написания автоматизированных тестов.

Поэтапное создание программы

В качестве цели тестирования разработаем простой класс `Calculator`, который умеет считать сумму двух чисел.

Автоматизированные тесты основаны на простом принципе сравнении результатов:

- Ожидаемый результат (expected result) – то, что мы ожидаем получить, например основываясь на требованиях к работе программы. Из школьного курса математики, мы знаем, что `2 + 2 = 4`, следовательно это ожидаемый результат;
- Фактический результат (actual result) – то, что мы получаем фактически, то есть в качестве результата работы программы, в данном случае метода `sum()`;

Если фактический результат совпадает с ожидаемым, тогда программа работает верно.

В программировании можно создавать тестовые сценарии, в рамках которых сравниваются результаты. Сравнение результатов также называют `assertion`. Как правило, все тестовые сценарии программируются в тестовых классах, которые дублируют название тестируемого класса, но к нему добавляется суффикс `Test`, например `CalculatorTest`.

Шаг 1.

Создадим класс `Calculator`:

```
public class Calculator {

    public int sum(int a, int b) {
        return a + b;
    }

}
```

Шаг 2.

Создадим класс `CalculatorTest`, в котором разработаем вспомогательный метод `checkThatEqual` для проверки результатов на равенство:

```
public class CalculatorTest {

    public void checkThatEqual(int expected, int actual, String testDescription) {
        if (expected == actual) {
            System.out.println(testDescription + " has passed!");
        } else {
            System.out.println(testDescription + " has failed!");
            System.out.println("Expected '" + expected + "' but was '" + actual + "'");
        }
    }

}
```

Если ожидаемый и фактический результат совпадают, то в консоль выведется текст с описанием тестового сценария и сообщение о том, что тейст успешно пройдет. В противном случае, мы увидим сообщение о проваленом тесте и информацию о том, что ожидалось, но было получено фактически.

‼️ **Примечание**: Для сравнение содержимого строк, вместо `==` необходимо использовать метод `.equals()`, например `expected.equals(actual)`.

Шаг 3.

Разработам классический тестовый сценарий. Отдельный метод считается тестовым сценарием, в котором проверяется та или иная часть работы класса:

```
public class CalculatorTest {

    public void test1() {
        String testDescription = "Should correctly sum two integers";

        Calculator victim = new Calculator();

        int expectedResult = 4;
        int actualResult = victim.sum(2, 2);

        checkThatEqual(expectedResult, actualResult, testDescription);
    }

    public void checkThatEqual(int expected, int actual, String testDescription) {
        if (expected == actual) {
            System.out.println(testDescription + " has passed!");
        } else {
            System.out.println(testDescription + " has failed!");
            System.out.println("Expected '" + expected + "' but was '" + actual + "'");
        }
    }

}
```

Обратите внимание, что в рамках метода создается объект класса `Calculator` с названием `victim`. Это один из подходов к наименованию класса, который мы хотим протестировать.

Шаг 4.

Для того, чтобы запустить тесты, необходимо запустить тестовые сценарии. Для этого в методе `main()` создадим экземпляр класса самого себя и вызовем метод – тестовый сценарий:

```
public class CalculatorTest {

    public static void main(String[] args) {

        CalculatorTest testRunner = new CalculatorTest();
        testRunner.test1();

    }

    public void test1() {
        String testDescription = "Should correctly sum two integers";

        Calculator victim = new Calculator();

        int expectedResult = 4;
        int actualResult = victim.sum(2, 2);

        checkThatEqual(expectedResult, actualResult, testDescription);
    }

    public void checkThatEqual(int expected, int actual, String testDescription) {
        if (expected == actual) {
            System.out.println(testDescription + " has passed!");
        } else {
            System.out.println(testDescription + " has failed!");
            System.out.println("Expected '" + expected + "' but was '" + actual + "'");
        }
    }

}
```

Результат работы программы:

> Should correctly sum two integers has passed!

Рекомендации:

- Запустить программу и сравнить результаты;
- Попробовать изменить логику в подсчете суммы и сравнить результат работы теста;
- Написать дополнительные тесты с другими данными;