МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федерально автономное бюджетное образовательное учреждение высшего образования

«Севастопольский государственный университет»

кафедра Информационных систем

Куркчи Ариф Эрнестович

Институт информационных технологий и управления в технических системах

курс 3 группа ИС/б-31-о

09.03.02 Информационные системы и технологии (уровень бакалавриата)

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Тестирование программного обеспечения»

на тему «Исследование способов модульного тестирования программного обеспечения»

Отметка о зачете \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_

(дата)

Руководитель практикума

ст. преподаватель   В. А. Строганов

(должность) (подпись) (инициалы, фамилия)

Севастополь 2016

1. ЦЕЛЬ РАБОТЫ

Исследовать основные подходы к модульному тестированию программного обеспечения. Приобрести практические навыки составления модульных тестов для объектно-ориентированных программ.

1. ПОСТАНОВКА ЗАДАЧИ

Тестируемый класс Stock должен предоставлять возможность хранения некоторого множества элементов класса Item, добавления их на склад и их изъятия.

Тестовый случай охватывает заполнение склада с последующей проверкой группировки элементов и проверку корректного изъятия элемента.

1. ХОД РАБОТЫ
2. Исходный код класса Stock

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | package ru.justnero.study.sevsu.pptc; | | 2 |  | | 3 | import java.util.HashMap; | | 4 | import java.util.Map; | | 5 |  | | 6 | public class Stock { | | 7 |  | | 8 | private final *Map<Integer, Item>* items; | | 9 | private final *Report* report; | | 10 |  | | 11 | private static final *Map<String, Stock>* instance = new *HashMap<>*(); | | 12 |  | | 13 | private Stock(*String* *reportName*) { | | 14 | items  = new *HashMap<>*(); | | 15 | report = new *Report*(reportName); | | 16 | } | | 17 |  | | 18 | public static *Stock* getInstance() { | | 19 | return getInstance("Default"); | | 20 | } | | 21 |  | | 22 | public static *Stock* getInstance(*String* *name*) { | | 23 | *Stock* inst = instance.get(name); | | 24 | if(inst == null) { | | 25 | inst = new *Stock*(name+" Stock report"); | | 26 | instance.put(name, inst); | | 27 | } | | 28 | return inst; | | 29 | } | | 30 |  | | 31 | public *int* getSize() { | | 32 | return items.size(); | | 33 | } | | 34 |  | | 35 | public *Stock* addItem(final *Item* *item*) { | | 36 | *Item* cur = items.get(item.getId()); | | 37 | if(cur == null) { | | 38 | items.put(item.getId(), item); | | 39 | report.addLine("Added new item ", item); | | 40 | } else { | | 41 | cur.setQuantity(cur.getQuantity() + item.getQuantity()); | | 42 | report.addLine("Added item ", cur); | | 43 | } | | 44 | return this; | | 45 | } | | 46 |  | | 47 | public *Item* getItem(*int* *id*) { | | 48 | *Item* item = items.get(id); | | 49 | if(item == null) { | | 50 | report.addLine("Item ", id, " not found"); | | 51 | } else { | | 52 | report.addLine("Item found ", item); | | 53 | } | | 54 | return items.get(id); | | 55 | } | | 56 |  | | 57 | public *Stock* removeItem(final *int* *id*) { | | 58 | *Item* item = items.get(id); | | 59 | if(item == null) { | | 60 | report.addLine("Item not removed ", id); | | 61 | } else { | | 62 | items.remove(id); | | 63 | report.addLine("Item removed ", item); | | 64 | } | | 65 | return this; | | 66 | } | | 67 |  | | 68 | public *Item* takeItem(final *int* *id*, final *int* *quantity*) { | | 69 | *Item* cur = items.get(id); | | 70 | *Item* item = null; | | 71 | if(cur != null) { | | 72 | if(cur.getQuantity() >= quantity) { | | 73 | item = new *Item*(id, cur.getName(), cur.getPrice(), quantity); | | 74 | if(cur.getQuantity() == quantity) { | | 75 | items.remove(id); | | 76 | report.addLine("Item fully taken ", item); | | 77 | } else { | | 78 | cur.setQuantity(cur.getQuantity() - quantity); | | 79 | report.addLine("Item partly taken ", item, " left ",cur.getQuantity()); | | 80 | } | | 81 | } else { | | 82 | report.addLine("Item can`t be taken ", cur, " need ", quantity - cur.getQuantity(), " more"); | | 83 | } | | 84 | } | | 85 | return item; | | 86 | } | | 87 |  | | 88 | public *Report* createReport() { | | 89 | return report; | | 90 | } | | 91 |  | | 92 | } | |  |

1. Исходный код тестового класса StockTest

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | package ru.justnero.study.sevsu.pptc; | | 2 |  | | 3 | import org.junit.After; | | 4 | import org.junit.Before; | | 5 | import org.junit.Test; | | 6 |  | | 7 | import java.math.BigInteger; | | 8 | import java.util.Random; | | 9 |  | | 10 | import static *org.hamcrest.CoreMatchers*.\*; | | 11 | import static *org.junit.Assert*.\*; | | 12 |  | | 13 | public class StockTest { | | 14 |  | | 15 | private *String* name; | | 16 | private *Stock* stock; | | 17 |  | | 18 | *@Before* | | 19 | public *void* setUpBeforeClass() throws *Exception* { | | 20 | name = new *BigInteger*(130, new *Random*()).toString(32); | | 21 | stock = *Stock*.getInstance(name); | | 22 | stock   .addItem(new *Item*(1,      "Монитор", 149.99D, 20)) | | 23 | .addItem(new *Item*(2,   "Клавиатура",  49.99D, 55)) | | 24 | .addItem(new *Item*(3,         "Мышь",  29.99D, 50)) | | 25 | .addItem(new *Item*(4,       "Корпус",  99.99D, 10)) | | 26 | .addItem(new *Item*(5,    "Процессор", 199.99D,  5)) | | 27 | .addItem(new *Item*(6, "Жёсткий диск", 129.99D, 15)); | | 28 | } | | 29 |  | | 30 | *@Test* | | 31 | public *void* fill() throws *Exception* { | | 32 | assertThat(stock.getSize(), is(6)); | | 33 | assertEquals(stock.getItem(5).getName(), "Процессор"); | | 34 | } | | 35 |  | | 36 | *@Test* | | 37 | public *void* modify() throws *Exception* { | | 38 | assertThat(stock.takeItem(4, 1).getQuantity(), is(1)); | | 39 | assertNull(stock.takeItem(4, 10)); | | 40 | assertThat(stock.getItem(4).getQuantity(), is(9)); | | 41 | } | | 42 |  | | 43 | *@Test* | | 44 | public *void* report() throws *Exception* { | | 45 | fill(); | | 46 | modify(); | | 47 |  | | 48 | *Report* report = stock.createReport(); | | 49 |  | | 50 | report.print(); | | 51 | assertThat(report.getSize(), is(10)); | | 52 | } | | 53 |  | | 54 | *@After* | | 55 | public *void* tearDownAfter() throws *Exception* { | | 56 | *System*.out.println(); | | 57 | } | | 58 |  | | 59 | } | |  |

В тестовом случае новый экземпляр класса Stock заполняется 6ю типами элементов определённого количества, после чего тестовый метод fill проверяет количество различных элементов, число которых должно быть равно 6, а элементом под ID 5 должен быть «Процессор».

Второй тестовый случай производит изъятие одного элемента под ID 4, после чего их количество должно уменьшится с 10 до 9. При попытке изъять 10 элементов под ID 4 ожидается null в результате.

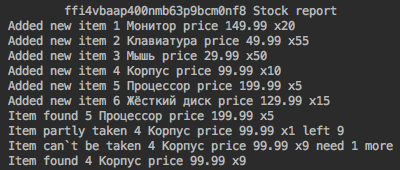


Рисунок 1 – Результат выполнения теста

ВЫВОДЫ

В ходи данной лабораторной работы были изучены подходы к модульному тестированию ПО. Был написан модульный тест по заданному тестовому случаю к одному из классов проекта по дисциплине «ТСПП». В результате выполнения теста была подтверждена правильность написания отдельного модуля, представленного в виде класса Stock.