**中国矿业大学计算机学院**

**2014 级本科生课程报告**

课程名称 软件测试技术

报告时间 2017年7月3日

学生姓名 赵勤博

学 号 08143427

专 业 计算机科学与技术

任课教师 王荣存

任课教师评语

任课教师评语（①对课程基础理论的掌握；②对课程知识应用能力的评价；③对课程报告相关实验、作品、软件等成果的评价；④课程学习态度和上课纪律；⑤课程成果和报告工作量；⑥总体评价和成绩；⑦存在问题等）：

成 绩： 任课教师签字：

年 月 日

摘 要

Selenium 是web自动化测试工具集，包括IDE、Grid、RC（selenium 1.0）、WebDriver（selenium 2.0）等。本文采用Selenium 2，又名 WebDriver对12306注册系统进行黑盒测试，发现网站现存的问题。本文采用白盒测试对Vector类进行测试，深入了解了单元测试的相关方法和对该类的理解。

**关键字：** Selenium 2；黑盒测试；12306；白盒测试

ABSTRACT

Selenium is a collection of Web automated testing tools, including IDE, Grid, RC (selenium 1), WebDriver (selenium 2), and so on. In this paper, Selenium 2, also called WebDriver, is used to test the black box of the 12306 registration system, and the existing problems of the website are found. In this paper, white box testing is used to test the Vector class, and the method of unit testing and the understanding of this class are deeply understood.

**KEYWORDS:** Selenium 2; black box testing; 12306; white box testing

目 录

[一 黑盒测试 1](#_Toc486866690)

[1.1Selenium简介 1](#_Toc486866691)

[1.2测试环境 1](#_Toc486866692)

[1.3测试过程 2](#_Toc486866693)

[1.3.1基础测试等价类 2](#_Toc486866694)

[1.3.2设计基础测试用例 3](#_Toc486866695)

[1.3.3 基础测试用例测试结果 5](#_Toc486866696)

[1.3.4学生类型测试等价类 6](#_Toc486866697)

[1.3.5设计学生类型测试用例 7](#_Toc486866698)

[1.3.6学生类型测试用例测试结果 8](#_Toc486866699)

[1.3.7证件类型测试等价类 9](#_Toc486866700)

[1.3.8设计证件类型测试用例 9](#_Toc486866701)

[1.3.9证件类型测试用例测试结果 11](#_Toc486866702)

[二 Vector类单元测试 12](#_Toc486866703)

[2.1源码分析 12](#_Toc486866704)

[2.1.1Vector简介 12](#_Toc486866705)

[2.1.2Vector数据结构 12](#_Toc486866706)

[2.2单元测试 12](#_Toc486866707)

[2.2.1准备阶段 12](#_Toc486866708)

[2.2.2构造方法的测试 13](#_Toc486866709)

[2.2.3 ensureCapacity方法测试 13](#_Toc486866710)

[2.2.4 Size()方法测试 17](#_Toc486866711)

[2.2.5 contains(Object elem) 方法测试 17](#_Toc486866712)

[2.2.6 indexOf(Object elem) 方法测试 17](#_Toc486866713)

[2.2.7 indexOf(Object elem, int index) 方法测试 17](#_Toc486866714)

[2.2.8setElementAt(E obj, int index) 方法测试 18](#_Toc486866715)

[2.2.9setElementAt(E obj, int index) 方法测试 18](#_Toc486866716)

[2.2.10removeElementAt(int index) 方法测试 19](#_Toc486866717)

[2.2.11addElement(E obj) 方法测试 19](#_Toc486866718)

[2.2.12removeElement(Object obj) 方法测试 20](#_Toc486866719)

[2.2.13get(int index) 方法测试 20](#_Toc486866720)

[2.3变异测试 20](#_Toc486866721)

[2.3.1 modCount- -变异 21](#_Toc486866722)

[2.3.2 j变异 21](#_Toc486866723)

[2.4测试结果 22](#_Toc486866724)

[2.4.1单元测试 22](#_Toc486866725)

[2.4.2变异测试结果 23](#_Toc486866726)

[三 总结 23](#_Toc486866727)

[参考文献 24](#_Toc486866728)

# 一 黑盒测试

## 1.1Selenium简介

Selenium也是一个用于Web应用程序测试的工具。Selenium测试直接运行在浏览器中，就像真正的用户在操作一样。支持的浏览器包括IE、Mozilla Firefox、Mozilla Suite等。这个工具的主要功能包括：测试与浏览器的兼容性——测试你的应用程序看是否能够很好得工作在不同浏览器和操作系统之上。测试系统功能——创建衰退测试检验软件功能和用户需求。支持自动录制动作和自动生成。Net、Java、Perl等不同语言的测试脚本。Selenium 是ThoughtWorks专门为Web应用程序编写的一个验收测试工具。

Selenium 2，又名 WebDriver，它的主要新功能是集成了 Selenium 1.0 以及 WebDriver（WebDriver 曾经是 Selenium 的竞争对手）。也就是说 Selenium 2 是 Selenium 和 WebDriver 两个项目的合并，即 Selenium 2 兼容 Selenium，它既支持 Selenium API 也支持 WebDriver API。 WebDriver是一个用来进行复杂重复的web自动化测试的工具。意在提供一种比Selenium1.0更简单易学，有利于维护的API。它没有和任何测试框架进行绑定，所以他可以很好的在单元测试和main方法中调用。一旦创建好一个Selenium工程，你马上会发现WebDriver和其他类库一样：它是完全独立的，你可以直接使用而不需要考虑其他配置，这个Selenium RC是截然相反的。

Webdriver的优点：当这两个框架被合并后，一个框架的缺陷被另一个框架所弥补。WebDriver对浏览器的支持需要对应框架开发工程师做对应的开发；同样Selenium必须操作真实浏览器，但是WebDriver可以HTML unit Driver来模拟浏览器，在内存中执行用例，更加的轻便。Selenium1.0解决了自动化测试中的一些常见问题，WebDriver更好的解决了沙箱限制。WebDriver不支持并行，但是Selenium Grid解决了这个问题。

## 1.2测试环境

1）Java 1.7

2）selenium-java 2.25.0

3）eclipse Luna Service Release 2 (4.4.2)

4）junit 4.11

5）maven 4.0

6）chromedriver 2.30

7）chrome浏览器59.0.3071.109

## 1.3测试过程

### 1.3.1基础测试等价类

表1.1 基础测试等价类表

|  |  |  |
| --- | --- | --- |
| 输入条件 | 有效等价类 | 无效等价类 |
| 用户名 | (1)长度6-30位  (2)字母、数字、“\_”组合  (3)字母开头  (4)用户名未被占用 | (16)长度<6  (17)长度>30  (18)不是以字母开头  (19)组合中存在除字母、数字和“\_”以外其它特殊符号  (20)用户名被占用  (21)空 |
| 登录密码 | (5)长度6-20位  (6)字母、数字、“\_”中的两种或两种以上 | (22)长度<6  (23)长度>20  (24)组合中存在除字母、数字以外其它符号  (25)只包含字母  (26)只包含数字  (27)只包含下划线  (28)空 |
| 确认密码 | (7)与密码相同 | (29)与密码不一致  (30)空 |
| 姓名 | (8)符合姓名填写规则 | (31)字符长度<3  (32)字符长度>30  (33)组合包含除英文、汉字、“.”、“·”以外特殊字符  (34)空 |
| 证件类型 | (9)二代身份证或港澳通行证或台湾通行证或护照 |  |
| 邮箱 | (10) 符合正则表达式/^([a-zA-Z0-9\_-])+@([a-zA-Z0-9-])+(.[a-zA-Z0-9])+/  (11)空 | (35)不符合正则表达式/^([a-zA-Z0-9\_-])[+@([a-zA-Z0-9-])+(.[a-zA-Z0-9])+/](mailto:+@([a-zA-Z0-9-])+(.[a-zA-Z0-9])+/)  (36)已被注册 |
| 手机号码 | (12)有效的格式；  (13)长度为11位 | (37) 已被注册  (38)长度<11  (39)长度>11  (40)无效的格式；  (41) 空 |
| 旅客类型 | (14)成人或儿童或伤残军人、伤残人民警察或学生 |  |
| 同意遵守条款 | (15)勾选 | (42)未勾选 |

### 1.3.2设计基础测试用例

表1.2 邮箱为空的通过测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 输入条件 | 输入 | 覆盖等价类 | 预期输出 |
| 用户名 | ks\_22222 | 1、2、3、4、5、6、7、8、9、11、12、13、14、15、61 | 出现完成注册按钮 |
| 登陆密码 | aaa123 |
| 确认密码 | aaa123 |
| 姓名 | fla |
| 证件类型 | 二代身份证 |
| 证件号码 | 440901197502108337（未被注册过） |
| 邮箱 |  |
| 手机号码 | 18311111111  （未被注册过） |
| 旅客类型 | 成人 |

表1.3 邮箱不为空的通过测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 输入条件 | 输入 | 覆盖等价类 | 预期输出 |
| 用户名 | ks\_22222 | 1、2、3、4、5、6、7、8、9、11、12、13、14、15、61 | 出现完成注册按钮 |
| 登陆密码 | aaa123 |
| 确认密码 | aaa123 |
| 姓名 | fla |
| 证件类型 | 二代身份证 |
| 证件号码 | 440901197502108337（未被注册过） |
| 邮箱 | 1-1@cn.com |
| 手机号码 | 18311111111  （未被注册过） |
| 旅客类型 | 成人 |

根据测试2修改的测试用例，只更改其中的一项用于覆盖无效等价类，详细输入参照程序Strings.java。

表1.4 用户名无效等价类覆盖测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 用户名 | 覆盖无效等价类 | 预期输出 |
| 3 | ks | 16 | 用户名长度不能少于6个字符！ |
| 4 | 123456 | 18 | 用户名只能由字母、数字和\_组成,须以字母开头! |
| 5 | ks#23343 | 19 | 用户名只能由字母、数字和\_组成,须以字母开头! |
| 6 | mysouldream | 20 | 该用户名已经占用，请重新选择用户名！ |
| 7 |  | 21 | 请输入用户名！ |

表1.5 登录密码无效等价类覆盖测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 登录密码 | 覆盖无效等价类 | 预期输出 |
| 8 | s1234 | 22 | 密码长度不能少于6个字符！ |
| 9 | s12345678901234567890 | 23 | 密码长度大于20个字符 |
| 10 | aaa@123 | 24 | 格式错误，必须且只能包含字母，数字，下划线中的两种或两种以上！ |
| 11 | aaaaaa | 23 | 格式错误，必须且只能包含字母，数字，下划线中的两种或两种以上！ |
| 12 | 111111 | 24 | 格式错误，必须且只能包含字母，数字，下划线中的两种或两种以上！ |
| 13 | \_\_\_\_\_\_ | 25 | 格式错误，必须且只能包含字母，数字，下划线中的两种或两种以上！ |
| 14 |  | 28 | 请输入密码！ |

表1.6 确认密码无效等价类覆盖测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 确认密码 | 覆盖无效等价类 | 预期输出 |
| 15 | aaa1234 | 29 | 确认密码与密码不同！ |
| 16 |  | 30 | 请输入确认密码！ |

表1.7 姓名无效等价类覆盖测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 姓名 | 覆盖无效等价类 | 预期输出 |
| 17 | jf | 31 | 允许输入的字符串在3-30个字符之间！ |
| 18 | j.flaj.flaj.flaj啊啊啊啊啊啊啊啊啊啊啊 | 32 | 允许输入的字符串在3-30个字符之间！ |
| 19 | j#fla | 33 | 姓名只能包含中文或者英文，如有生僻字或繁体字参见姓名填写规则进行填写！ |
| 20 |  | 34 | 请输入您的姓名！ |

表1.8 邮箱无效等价类覆盖测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 邮箱 | 覆盖无效等价类 | 预期输出 |
| 21 | 52@.com | 35 | 请输入有效的电子邮件地址！ |
| 22 | 583053544@qq.com | 36 | 该邮箱已被注册，请更换邮箱。 |

表1.9 邮箱无效等价类覆盖测试用例

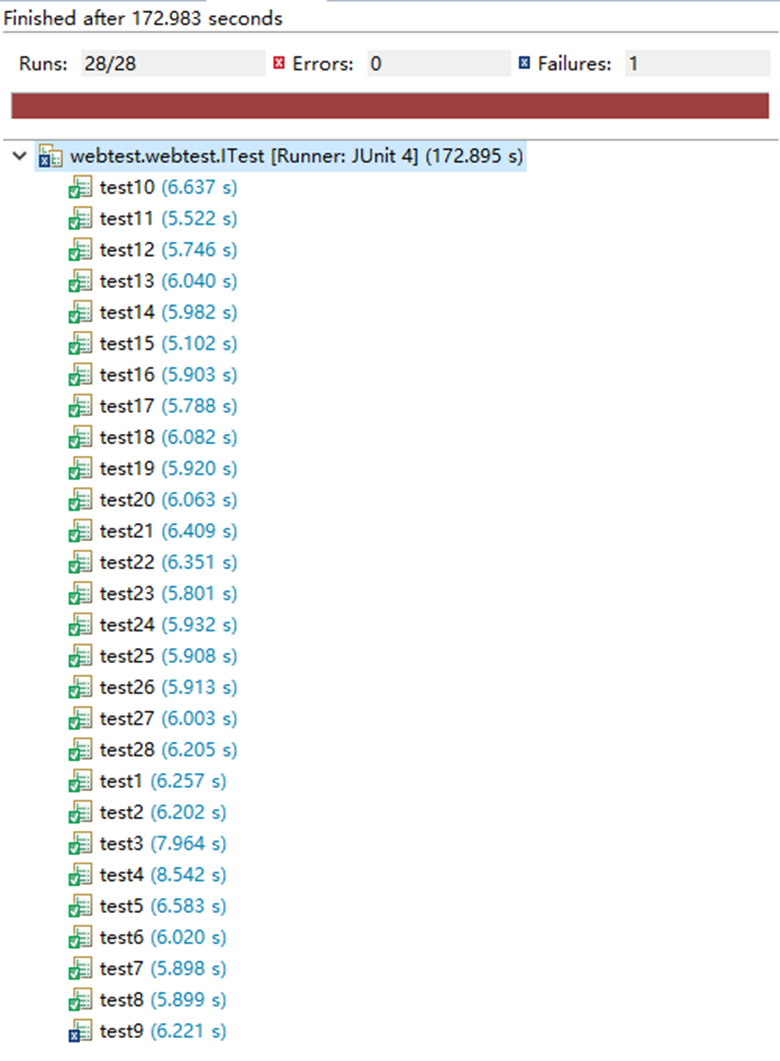
|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 手机号码 | 覆盖无效等价类 | 预期输出 |
| 23 | 18361221565（已被注册） | 37 | 包含 您输入的手机号码已被其他注册用户 的提示 |
| 24 | 1836122156 | 38 | 您输入的手机号码不是有效的格式！ |
| 25 | 183612215655 | 39 | 您输入的手机号码不是有效的格式！ |
| 26 | 039-61221565 | 40 | 您输入的手机号码不是有效的格式！ |
| 27 |  | 41 | 请输入手机号码！ |

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 同意遵守条款 | 覆盖无效等价类 | 预期输出 |
| 28 | 未勾选 | 42 | 请确定服务条款！ |

### 1.3.3 基础测试用例测试结果

测试9发现，当密码超过20位时，系统并不会出现预期输出，并不会提示密码超过20位，且能注册成功，与提示6-20位不符合。

图1.1 基础测试用例测试结果



### 1.3.4学生类型测试等价类

表1.9学生类型测试等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 前置条件 | 输入条件 | 有效等价类 | 无效等价类 |
| 选择旅客类型为学生 | 学校所在省份 | (43)选择省份 | 无 |
| 学校名称 | (44)选择学校 | (55)空 |
| 院系 | (45)由中文、英文、数字组成  (46)空 | (56)含中文、英文、数字以外的字符 |
| 班级 | (47)由中文、英文、数字组成  (48)空 | (57)含中文、英文、数字以外的字符 |
| 学号 | (49)由中文、英文、数字组成 | (58)含中文、英文、数字以外的字符  (59)空 |
| 学制 | (50)选择学制 |  |
| 入学年份 | (51)选择入学年份 |  |
| 优惠卡号 | (52)由中文、英文、数字组成  (53)为空 | (60)含中文、英文、数字以外的字符 |
| 优惠区间 | (54)任意输入 |  |

### 1.3.5设计学生类型测试用例

设计通过注册的测试用例测试29。

表1.10 学生类型通过测试的测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 输入条件 | 输入 | 覆盖等价类 | 预期输出 |
| 用户名 | ks\_22222 | 1、2、3、4、5、6、7、8、9、11、12、13、14、15、43、44、45、47、49、50、51、52、54、61 | 出现完成注册按钮 |
| 登陆密码 | aaa123 |
| 确认密码 | aaa123 |
| 姓名 | fla |
| 证件类型 | 二代身份证 |
| 证件号码 | 440901197502108337（未被注册过） |
| 邮箱 | 1-1@cn.com |
| 手机号码 | 18311111111  （未被注册过） |
| 旅客类型 | 成人 |
| 学校名称 | 中国矿业大学 |
| 院系 | 计算机学院 |
| 班级 | 计科3班 |
| 学号 | 08143427 |
| 学制 | 4 |
| 入学年份 | 2014 |
| 优惠卡号 |  |
| 优惠区间1 | 徐州 |
| 优惠区间2 | 兰州 |

根据测试29修改的测试用例，只更改其中的一项用于覆盖无效等价类，详细输入参照程序Strings.java。

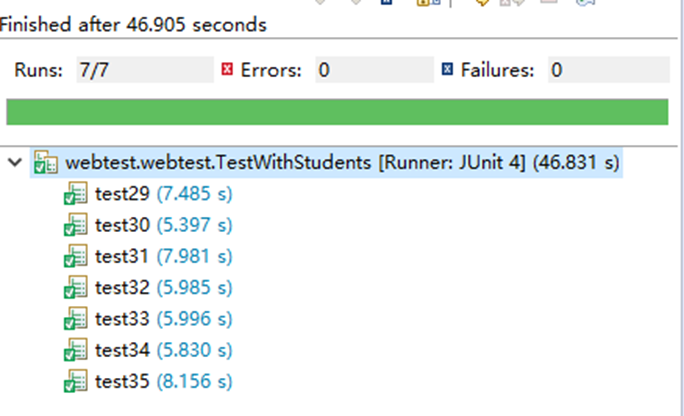
表1.11学生类型无效等价类覆盖测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 输入条件 | 输入数据 | 覆盖无效等价类 | 预期输出 |
| 30 | 学校名称 |  | 55 | 请选择学校名称！ |
| 31 | 院系 | 计算机学院# | 56 | 院系只能包含中文、英文、数字！ |
| 32 | 班级 | 计科%班 | 57 | 班级只能包含中文、英文、数字！ |
| 33 | 学号 |  | 58 | 请输入学号 |
| 34 | 学号 | 081434& | 59 | 学号只能包含中文、英文、数字！ |
| 35 | 优惠卡号 | 1# | 60 | 优惠卡号只能包含中文、英文、数字！ |

### 1.3.6学生类型测试用例测试结果

测试结果与预期输出一致。

图 1.2学生类型测试用例测试结果



### 1.3.7证件类型测试等价类

表1.12 证件类型测试等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 前置条件 | 输入条件 | 有效等价类 | 无效等价类 |
| 选择证件类型为二代身份证 | 证件号码 | (61)有效的身份证号 | (74)无效的身份证号  (75)已被注册  (76)空 |
| 选择证件类型为港澳通行证 | 证件号码 | (62)以“H”或“M” 或“h”或“m”开头。  (63)首字母接8位或10位数字 | (77)不是以“M”或“H” 或“h”或“m”开头  (78)数字长度不是8或10位  (79)后8位中含有非数字  (80)空 |
| 性别 | (64)选择“男”或“女” | (81)未选择 |
| 出生日期 | (65)选择出生日期 |  |
| 选择证件类型为台湾通行证 | 证件号码 | (66)由8位或10位数字组成 | (82)长度不是8或10位  (83)含数字以外的符号、字母  (84)空 |
| 性别 | (67)选择“男”或“女” | (85)未选择性别 |
| 出生日期 | (68)选择出生日期 | 无 |
| 选择证件类型为护照 | 证件号码 | (69)长度5～17位  (70)由字母、数字组成 | (86) 长度<5  (87)长度>17空  (88)含字母、数字以外的汉字或字符  (89)空 |
| 性别 | (71)选择“男”或“女” | (90)未选择性别 |
| 国家 | (72)选择国家 |  |
| 出生日期 | (73)选择出生日期 |  |

### 1.3.8设计证件类型测试用例

根据测试2修改的测试用例，只更改其中的一项用于覆盖无效等价类，详细输入参照程序Strings.java。

表1.13 证件类型测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 输入条件 | 输入数据 | 覆盖无效等价类 | 预期输出 |
| 36 | 证件号码（身份证） | 410112111326561255 | 74 | 请正确输入18位的身份证号！ |
| 37 | 证件号码（身份证） | 420800198404138910 | 75 | 包含 该证件号码已被注册 |
| 38 | 证件号码（身份证） |  | 76 | 请输入证件号码！ |
| 39 | 证件号码（港澳通行证） | H123456789 | 78 | 请输入有效的港澳居民通行证号码！ |
| 40 | 证件号码（港澳通行证） | 12345678 | 77 | 请输入有效的港澳居民通行证号码！ |
| 41 | 证件号码（港澳通行证） | H1234567x | 79 | 请输入有效的港澳居民通行证号码！ |
| 42 | 证件号码（港澳通行证） |  | 80 | 请输入证件号码！ |
| 43 | 证件号码（台通行证） | 123456789 | 82 | 请输入有效的台湾居民通行证号码！ |
| 44 | 证件号码（台通行证） | 3433456# | 83 | 输入的证件编号中包含中文信息或特殊字符！ |
| 45 | 证件号码（台通行证） |  | 84 | 请输入证件号码！ |
| 46 | 证件号码（护照） | 1234 | 86 | 请输入有效的护照号码！ |
| 47 | 证件号码（护照） | 12345678901234567890 | 87 | 请输入有效的护照号码！ |
| 48 | 证件号码（护照） | 12345# | 88 | 输入的证件编号中包含中文信息或特殊字符！ |
| 49 | 证件号码（护照） |  | 89 | 请输入证件号码！ |

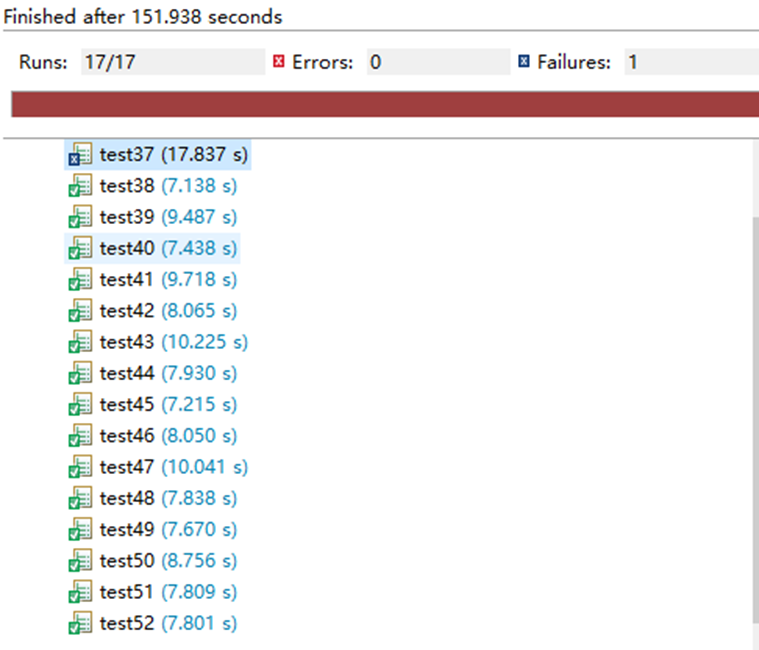
表1.14 证件类型测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 前置条件 | 输入条件 | 输入数据 | 覆盖无效等价类 | 预期输出 |
| 50 | 选择港澳通行证 | 性别 |  | 81 | 请选择性别 |
| 51 | 选择台湾通行证 | 性别 |  | 85 | 请选择性别 |
| 52 | 选择护照 | 性别 |  | 90 | 请选择性别 |

### 1.3.9证件类型测试用例测试结果

测试37为身份证已注册的测试方法，测试发现，在仅有身份证号相同时并不会触发身份证已被注册的提示，只有当输入的姓名和身份证号一致时，才会提示身份证已被注册错误。

图 1.3证件类型测试用例测试结果



# 二 Vector类单元测试

## 2.1源码分析

### 2.1.1Vector简介

Vector 是矢量队列，它是JDK1.0版本添加的类。继承于AbstractList，实现了List, RandomAccess, Cloneable这些接口。

Vector 继承了AbstractList，实现了List；所以，它是一个队列，支持相关的添加、删除、修改、遍历等功能。

Vector 实现了RandmoAccess接口，即提供了随机访问功能。RandmoAccess是java中用来被List实现，为List提供快速访问功能的。在Vector中，我们即可以通过元素的序号快速获取元素对象；这就是快速随机访问。

Vector 实现了Cloneable接口，即实现clone()函数。它能被克隆。

和ArrayList不同，Vector中的操作是线程安全的。

### 2.1.2Vector数据结构

Vector的数据结构包含了3个成员变量：elementData , elementCount， capacityIncrement。

(1) elementData 是"Object[]类型的数组"，它保存了添加到Vector中的元素。elementData是个动态数组，如果初始化Vector时，没指定动态数组的大小，则使用默认大小10。随着Vector中元素的增加，Vector的容量也会动态增长，capacityIncrement是与容量增长相关的增长系数，具体的增长方式参考源码分析中的ensureCapacity()函数。

(2) elementCount 是动态数组的实际大小。

(3) capacityIncrement 是动态数组的增长系数。如果在创建Vector时，指定了capacityIncrement的大小，则每次当Vector中动态数组容量增加时，增加的大小都是capacityIncrement。

## 2.2单元测试

### 2.2.1准备阶段

创建v0方便后续使用。Vector<Object> v0= null;。

添加@Rule注解，public ExpectedException thrown= ExpectedException.none();，可通过thrown测试带抛出信息的异常。

### 2.2.2构造方法的测试

本文认为，为验证构造函数的正确性，通常判断产生对象属性的值与预期是否相同。无参数的构造会向上补全默认参数，下面按参数由少至多分别给出测试方法。

（1）Vector()：无参数的构造函数默认容量为10，容量增长为0，即为每次增长一倍。

v0 = new Vector<Object>();

assertEquals(10, v0.elementData.length);

assertEquals(0, v0.capacityIncrement);

(2) Vector(int initialCapacity)：initialCapacity为初始容量，容量增长为0，即为每次增长一倍。

v0 = new Vector<Object>(5);

assertEquals(5, v0.elementData.length);

assertEquals(0, v0.capacityIncrement);

(3) Vector(int initialCapacity, int capacityIncrement)： initialCapacity为初始容量，capacityIncrement为容量增长。

v0 = new Vector<Object>(3,2);

assertEquals(3, v0.elementData.length);

assertEquals(2, v0.capacityIncrement);

（4）Vector(Collection<? extends E> c)：指定集合的Vector构造函数

Collection<Object> c = new ArrayList<Object>();

c.add(1);

v0 = new Vector<Object>(c);

assertEquals((int) Math.min((c.size() \* 110L) / 100,

Integer.MAX\_VALUE), v0.elementData.length);

assertEquals(0, v0.capacityIncrement);

（5）初始容量小于0异常

@Test(expected = IllegalArgumentException.class)

public void testVectorEx() {

v0 = new Vector<Object>(-1);

}

(6)杀死if (initialCapacity <= 0)变异

v0 = new Vector<Object>(0,0);

assertEquals(0, v0.elementData.length);

assertEquals(0, v0.capacityIncrement);

### 2.2.3 ensureCapacity方法测试

(1)源码分析

public void ensureCapacity(int minCapacity) {

modCount++;

ensureCapacityHelper(minCapacity);

}

// 确认“Vector容量”的帮助函数

private void ensureCapacityHelper(int minCapacity) {

int oldCapacity = elementData.length;

// 当Vector的容量不足以容纳当前的全部元素，增加容量大小。

// 若 容量增量系数>0(即capacityIncrement>0)，则将容量增大capacityIncrement

// 否则，将容量增大一倍。

if (minCapacity > oldCapacity) {

Object[] oldData = elementData;

int newCapacity = (capacityIncrement > 0) ? (oldCapacity + capacityIncrement)

: (oldCapacity \* 2);

if (newCapacity < minCapacity) {

newCapacity = minCapacity;

}

elementData = new Object[newCapacity];

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

elementData[i] = oldData[i];

}

// System.arraycopy(oldData, 0, elementData, 0, elementCount);

} }

（2）测试策略

组合覆盖要求设计足够多的测试用例，使得每个判定中条件结果的所有可能组合至少出现一次。

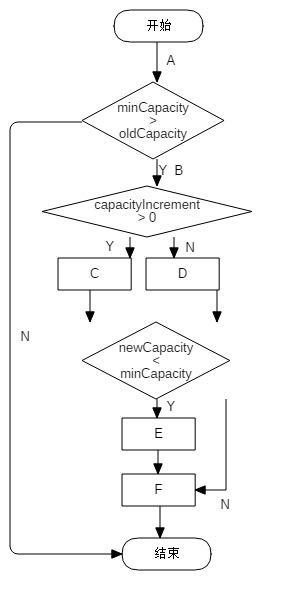


图2.1 ensureCapacityHelper方法流程图

表2.1组合覆盖表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| minCapacity | oldCapacity | Capacity  Increment | 判定1 | 判定2 | 判定3 | 预期结果 | 覆盖路径 |
| 4 | 1 | 2 | T | T | T | 4 | ABCEF |
| 3 | 1 | 2 | T | T | F | 3 | ABCF |
| 3 | 1 | 0 | T | F | T | 3 | ABDEF |
| 2 | 1 | 0 | T | F | F | 2 | ABDE |
| 0 | X | Y | F | - | - | X | A |

@Test

public void testEnsureCapacity1() {

v0 = new Vector<Object>(1,2);

v0.addElement(1);

v0.ensureCapacity(4);

assertEquals(4 , v0.elementData.length);

assertEquals(2, (vt.getSpecificedFieldObject(v0)));

}

@Test

public void testEnsureCapacity2() {

v0 = new Vector<Object>(1,2);

v0.ensureCapacity(3);

assertEquals(3, v0.elementData.length);

}

@Test

public void testEnsureCapacity3() {

v0 = new Vector<Object>(1,0);

v0.ensureCapacity(3);

assertEquals(3 , v0.elementData.length);

}

@Test

public void testEnsureCapacity4() {

v0 = new Vector<Object>(1,0);

v0.ensureCapacity(2);

assertEquals(2 , v0.elementData.length);

}

@Test

public void testEnsureCapacity5() {

v0 = new Vector<Object>(5,0);

v0.ensureCapacity(0);

assertEquals(5 , v0.elementData.length);

}

### 2.2.4 Size()方法测试

用于测试Size()方法，新创建的Vector大小为0。

v0 = new Vector<Object>();

assertEquals(0, v0.size());

### 2.2.5 contains(Object elem) 方法测试

用于测试contains(Object elem)方法，新建Vector不包含任意值，加入元素后包含。

v0 = new Vector<Object>();

assertFalse(v0.contains(1));

v0.addElement(2);

assertTrue(v0.contains(2));

### 2.2.6 indexOf(Object elem) 方法测试

用于测试indexOf(Object elem)方法，返回某元素的位置。

public void testindexOf()

{

v0 = new Vector<Object>();

assertEquals(-1, v0.indexOf(1));

v0.addElement(2);

assertEquals(0, v0.indexOf(2));

}

### 2.2.7 indexOf(Object elem, int index) 方法测试

用于测试indexOf(Object elem, int index)方法，返回index后某元素的位置。

public void testindexOf2()

{

v0 = new Vector<Object>();

assertEquals(-1, v0.indexOf(null,0));

v0.addElement(2);

assertEquals(-1, v0.indexOf(null,0));

v0.addElement(null);

assertEquals(1, v0.indexOf(null,0));

assertEquals(0, v0.indexOf(2,0));

assertEquals(-1, v0.indexOf(2,1));

}

### 2.2.8setElementAt(E obj, int index) 方法测试

用于测试setElementAt(E obj, int index)方法，在index设置对象。

@Test(expected = ArrayIndexOutOfBoundsException.class)

public void testsetElementAtEX()

{

v0 = new Vector<Object>();

v0.setElementAt(2, 0);

}

@Test

public void testsetElementAt()

{

v0 = new Vector<Object>();

v0.addElement(1);

v0.setElementAt(2, 0);

assertEquals(2, v0.elementData[0]);

}

### 2.2.9setElementAt(E obj, int index) 方法测试

用于测试setElementAt(E obj, int index)方法的异常。

setElementAt(E obj, int index) public void testremoveElementAt()

{

v0 = new Vector<Object>();

thrown.expect(ArrayIndexOutOfBoundsException.class);

thrown.expectMessage(" >= ");

v0.removeElementAt(0);

}

public void testremoveElementAt2()

{

v0 = new Vector<Object>();

thrown.expect(ArrayIndexOutOfBoundsException.class);

thrown.expectMessage("<0");

v0.removeElementAt(-1);

}

@Test

public void testremoveElementAt3()

{

v0 = new Vector<Object>();

v0.addElement(1);

v0.removeElementAt(0);

assertEquals(null, v0.elementData[0]);

assertEquals(2 , (vt.getSpecificedFieldObject(v0)));

v0.addElement(1);

v0.addElement(2);

v0.removeElementAt(0);

assertEquals(2, v0.elementData[0]);

assertEquals(5 , (vt.getSpecificedFieldObject(v0)));

}

### 2.2.10removeElementAt(int index) 方法测试

用于测试removeElementAt(int index)方法，移除index位置的元素。

public void testremoveElement()

{

v0 = new Vector<Object>();

assertFalse(v0.removeElement(1));

v0.addElement(1);

assertTrue(v0.removeElement(1));

assertEquals(4 , (vt.getSpecificedFieldObject(v0)));

}

### 2.2.11addElement(E obj) 方法测试

用于测试addElement(E obj)方法，用于添加元素。

public void testaddElement()

{

v0 = new Vector<Object>(0,0);

v0.addElement(1);

assertEquals(1, v0.elementData.length);

assertEquals(1, v0.elementData[0]);

assertEquals(1 , (vt.getSpecificedFieldObject(v0)));

}

### 2.2.12removeElement(Object obj) 方法测试

用于测试removeElement(Object obj)方法，用于移除元素。

public void testremoveElement()

{

v0 = new Vector<Object>();

assertFalse(v0.removeElement(1));

v0.addElement(1);

assertTrue(v0.removeElement(1));

assertEquals(4 , (vt.getSpecificedFieldObject(v0)));

}

### 2.2.13get(int index) 方法测试

用于测试get(int index)方法，用于获得元素。

@Test(expected = ArrayIndexOutOfBoundsException.class)

public void testgetEX()

{

v0 = new Vector<Object>();

v0.get(0);

}

@Test()

public void testget()

{

v0 = new Vector<Object>();

v0.addElement(1);

assertEquals(1, v0.get(0));;

}

## 2.3变异测试

变异测试，通过选择一些变异操作，并对于每一个可执行代码段依次把它们应用在源代码中。对程序使用变异操作的结果叫做一个突变异种。如果测试单元可以察觉到错误（即：一个测试失败了），那么就说该突变异种被杀害了。

### 2.3.1 modCount- -变异

杀死变异modCount--;的方法，通过反射获取对象父类的modCount属性，验证modCount的值变化是否正确。

public Object getSpecificedFieldObject(Vector<Object> v) {

Class<?> clazz = Vector.class.getSuperclass();

Object object = null;

try {

Field field = clazz.getDeclaredField("modCount");

field.setAccessible(true);

System.out.println();

object = field.get(v);

}

catch (NoSuchFieldException e) {

e.printStackTrace();

}

catch (SecurityException e) {

e.printStackTrace();

}

catch (IllegalArgumentException e) {

e.printStackTrace();

}

catch (IllegalAccessException e) {

e.printStackTrace();

}

return object;

}

创建一个对象VectorTest vt，可通过vt.getSpecificedFieldObject(v0)获取v0对象父类AbstractList的modCount属性，用于记录修改的次数。

### 2.3.2 j变异

源码语句117行int j = elementCount - index -1;，变异为int j = elementCount + index -1;和int j = elementCount - index +1;两种，因为j是内部变量，无法通过直接判断j值的方式杀死变异。故考虑根据产生的结果不同来杀死变异，j为elementData[]中元素移动的次数，删除某元素后只需要移动该元素index后的Object，但变异只会增加移动次数而并不会改变移动后elementData[]的内容，所以通过elementData[]的对比也无法杀死变异。考虑到Vector的容量，当程序想移动更多的步数，可能会超出Vector本身的范围，所以给elementData[]一个刚和满足Vector容量的值，若移动次数增加则会产生数组越界异常，若移动次数减少，则无法获得正确的结果。测试程序如下：

@Test

public void testremoveElementAt4()

{

v0 = new Vector<Object>(4,0);

v0.addElement(1);

v0.addElement(2);

v0.addElement(null);

v0.addElement(4);

int index=1;

int j= v0.elementCount-index-1;

Object[] o = {new Integer(1),null,new Integer(4)};

v0.removeElementAt(1);

//验证结果

for (int i = 0; i < v0.elementData.length; i++) {

if(i<v0.elementCount)

assertEquals(o[ i ], v0.elementData[i]);

else

assertEquals(null, v0.elementData[i]);

}

assertEquals(4, v0.elementData.length);

}

## 2.4测试结果

### 2.4.1单元测试



### 2.4.2变异测试结果



# 三 总结

通过对这门课的学习，我学习到了软件的基本测试方法，软件的测试不是在软件完成之后进行的，甚至在编码之前就要进行测试，有的程序开发过程就是测试驱动的。测试不仅仅是发现程序功能的错误，还有性能方面，可靠性，可扩展性等。

测试贯穿程序开发的始终，白盒测试，测试人员根据程序内部的构造，要求对被测程序的结构特性做到一定的覆盖，覆盖有很多类型，比如基本路径覆盖，条件组合覆盖，条件判定覆盖等，通过对junit测试框架的学习和目测平台的练习，测试能力得到了提升，但测试还要讲究效率，如何在短时间内获得较大的测试成果也是一个问题，在软件测试的比赛中因为时间不足没有取得满意的分数。但这对今后编程提供了很大的帮助。

通过本次课程报告实践了黑盒测试的流程，从等价类划分到测试用例设计，再到Selenium自动化测试工具，使我对自动化测试有了初步认识，让我直观的了解这一技术，把书本上的理论知识实践到项目中。与之前在实验课程中的性能测试，压力测试结合，使我对一个系统的评价有了更全面的认识。

我很高兴能接触这门课程，并希望这门课程能够提前到之前的学期中，软件测试不需要太多的基础知识，但对软件开发的帮助确是很大的。遗憾的是，对于集成测试和数据库测试还并没有实践过，只停留在理论阶段。希望课程以后授课中编码实践的环节能够更多。

最后感谢王老师的培养教育，王老师终认真负责地给予我们深刻而细致地指导，帮助我们开拓研究思路，精心点拨、热忱鼓励，这些使我倍感亲切。

# 参考文献

[1]宫云战. 软件测试教程[M]. 北京：机械工业出版社，2016，38-84

[2]赵卓. Selenium自动化测试指南[M]. 北京：人民邮电出版社，2013，72-80