DuangCalender 简易记事本

测试评估报告

版本 <1.1>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <2019/4/14> | <1.0> | <黑盒测试> | 侯哲宇，张啸，王力帆，汤志彪 |
| <2019/5/11> | <1.1> | <白盒测试> | 张啸 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 3

1.1 目的 3

1.2 范围 3

1.3 定义、首字母缩写词和缩略语 3

1.4 参考资料 3

1.5 概述 3

2. 测试结果摘要 3

3. 基于需求的测试覆盖 3

4. 基于代码的测试覆盖 3

5. 建议措施 3

6. 图 3

测试评估报告

# 简介

## 目的

根据需求说明书，从用户角度对程序进行黑盒测试，根据功能一一设计并执行测试用例。

经过对不同软件的不断筛选，本次测试我们选择使用“DuangCalender” 作为测试对象。第一次次测试目的在于了解黑盒测试的原理及方法。“DuangCalender”是一款用来进行日期查询与记事的一个程序，分为日历与日历记事本两部分，我们针对这两部分分别设计测试用例，在不涉及代码的前提下对软件的功能进行测试。最终的测试用例将采用等价类测试和边界类两种测试方法对该程序进行测试。本测试需要测试的功能有对特定日期的查询，以及对给定日期的备忘录进行存档、清空存档。

等价类是指某个输入域的子集合。在该子集合中，各个输入数据对于揭露程序中的错误都是等效的，具有等价特性。等价类划分可有两种不同的情况：有效等价类和无效等价类。有效等价类是指对于程序的需求规格说明来说是合理的，有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能（确认过程）。无效等价类是指是指对于程序的需求规格说明来说是不合理的，无意义的输入数据构成的集合。利用无效等价类可检验程序对于无效数据的异常处理能力（检验过程）。

等价类测试方法是把所有可能的输入数据，即程序的输入域划分成若干部分，然后从每一部分中选取少数有代表性的数据作为测试用例。使用等价类划分方法设计测试用例要经历划分等价类（列出等价类表）和选取测试用例两步，它将不能穷举的测试过程进行合理分类，从而保证设计出来的测试用例具有完整性和代表性。

第二次测试目的为了通过检查软件内部的逻辑结构，对软件中的逻辑路径进行覆盖测试;在程序不同地方设立检查点，检查程序的状态，以确定实际运行状态与预期状态是否一致

## 范围

主要对日志和日历记事本两部分进行功能测试。

第一次测试为黑盒测试，即检测“DuangCalender”的每个功能是否都能正常使用，所有的输入框与可点击的按键都需要接受功能性测试。

功能性测试所测试的功能有：

功能1：通过日历进行日期查询（上年、下年和上月、下月）

功能2：通过日历记事本提供输入、保存、显示、删除日志的功能

由于本次测试是只针对于程序功能的黑盒测试，不涉及程序的代码方面，因此如果在测试过程中发现程序中存在的问题，也无法通过修改代码的方式来解决问题。本次测试仅对“DuangCalender”的每个功能能否正常使用进行测试。

第二次为白盒测试，主要通过Calenderpad主程序 中的Calenderpad ， Month ，Notepad ， Year 四个类进行白盒测试。

## 定义、首字母缩写词和缩略语

## 参考资料

https://github.com/programmer-zhang/DuangCalender

## 概述

此次测试结果主要以黑盒测试，白盒测试的方法，对日历设置用例测试功能。

# 测试结果摘要

-

# 基于需求的测试覆盖

日历部分：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 等价类 | 测试用例 | 预期结果 |
| 1 | 年份输入为整数 | 输入年份为2020 | 成功接收 |
| 2 | 年份输入为字符 | 输入年份为a\_B | 年份仍旧为之前的年份 |
| 3 | 年份输入为小数 | 输入年份为21.5 | 年份仍旧为之前的年份 |
| 编号 | 边界值 | 测试用例 | 预期结果 |
| 4 | 年份 | 输入年份1000年 | 正常输出 |
| 5 | 年份 | 输入年份999年 | 年份不变 |
| 6 | 年份 | 在当前年份为1000年时点击上年 | 年份不变 |
| 7 | 年份 | 输入年份3000年 | 正常输出 |
| 8 | 年份 | 输入年份为3001年 | 年份不变 |
| 9 | 年份 | 在当前年份为3000年时点击下年 | 年份不变 |
| 10 | 月份 | 7月时点击上月 | 返回6月 |
| 13 | 月份 | 7月时点击下月 | 返回8月 |
| 11 | 月份 | 2000年1月 点击上月 | 返回2019年的12月 |
| 14 | 月份 | 2000年12月 点击下月 | 返回2019年的1月 |
| 12 | 月份 | 1000年1月点击上月 | 显示1000年12月 |
| 15 | 月份 | 3000年12月点击下月 | 显示3000年1月 |
| 16 | 日期 | 点击日历内空白日期 | 无反应 |
| 17 | 日期 | 点击2019年4月14日（有日志的日期） | 跳出2019年4月14日日志 |
| 18 | 日期 | 点击2019年4月20日（无日志的日期） | 右侧日志框内显示“无记录” |

记事本部分：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 等价类 | 测试用例 | 预期结果 |
| 19 | 存档信息至某一未存档的日期 | 鼠标选择2019年4月11日，在右侧输入框内输入信息“1”，点击“保存日志” | 日志保存成功 |
| 20 | 查看某一已存档的日期内保存的信息 | 在测试19的基础上，鼠标选择2019年4月11日，在右侧输入框中查看信息 | 4月11日日志显示结果为“1” |
| 21 | 查看某一未存档的日期内保存的信息 | 鼠标选择2019年3月11日，在右侧输入框中查看信息 | 3月11日日志显示结果为“无记录” |
| 22 | 修改某一已存档的日期内保存的信息 | 在测试19的基础上，鼠标选择2019年4月11日，将右侧输入框中的“1”改为“2”，点击“保存日志” | 4月11日志显示结果为“2” |
| 23 | 清空某一已存档的日期内保存的信息 | 在测试19的基础上，鼠标选择2019年4月11日，点击“删除日志” | 4月11日日志被删除再次查询为“无记录” |
| 24 | 清空某一未存档的日期内保存的信息 | 鼠标选择2019年3月11日，点击“删除日志” | 弹出窗口提示该日无日志 |

# 基于代码的测试覆盖

对于Calenderpad主程序 中的Calenderpad ， Month ，Notepad ， Year 四个类进行白盒测试中的语句覆盖测试：

Calenderpad类（主类）：

1.set\_calendar()测试：

public void set\_calendar(int year, int month) {

calendar.set(year, month - 1, 1);

week = calendar.get(Calendar.DAY\_OF\_WEEK) - 1;

if (month == 1 || month == 3 || month == 5 || month == 7

|| month == 8 || month == 10 || month == 12) {

daynum=31;

arraynum(week, 31);

} else if (month == 4 || month == 6 || month == 9 || month == 11) {

daynum=30;

arraynum(week, 30);

} else if (month == 2) {

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

daynum=29;

arraynum(week, 29);

} else {

daynum=28;

arraynum(week, 28);

}

}

}

测试用例：

大月31天测试 2019/5/31

小月30天测试 2019/4/30

平年二月测试 2019/2/28 1800/2/28

闰年二月测试 2016/2/29 2000/2/29

年份超出范围测试 4500/5/31 4500/4/30 4500/2/28

2.arraynum()测试：

public void arraynum(int week, int month) {

for (int i = week, n = 1; i < week + month; i++) {

showDay[i].setText("" + n);

if (n == day) {

showDay[i].setForeground(Color.black);

showDay[i].setBackground(Color.blue);

showDay[i].setFont(new Font("TimesRoman", Font.BOLD, 20));

} else {

showDay[i].setFont(new Font("TimesRoman", Font.BOLD, 12));

showDay[i].setBackground(Color.white);

showDay[i].setForeground(Color.black);

}

if (i % 7 == 6) //设置日期为周六显示为红色

{

showDay[i].setForeground(Color.red);

}

if (i % 7 == 0) //设置日期为周日显示为红色

{

showDay[i].setForeground(Color.red);

}

n++;

}

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

showDay[i].setText("");

}

for (int i = week + month; i < 42; i++) {

showDay[i].setText("");

}

}

测试用例：设置arraynum（3,31）

参数为3，blue 5月1日为目标日 为蓝色背景

参数为4，white 5月2日不是目标日 为白色背景

参数为6，red 周六红色字体

参数为7，red 周日红色字体

3.setYear():

public void setYear(int y) {

if(y>=1000 && y<=3000) {

year = y;

notepad.setYear(year);

}

else{

notepad.setYear(2019);

}

}

测试用例： 2000

4.setMonth（）：

public void setMonth(int m) {

month = m;

notepad.setMonth(month);

}

测试用例：10

# 5. setDay()：

public void setDay(int d) {

day = d;

notepad.setDay(day);

}

测试用例：30

Month类：新建“月”面板，添加“上月”和“下月”按钮

1. getMonth():

public int getMonth() {

return month;

}

测试用例：5

1. setMonth（）：

public void setMonth(int month) {

if (month <= 12 && month >= 1) {

this.month = month;

} else {

this.month = 1;

}

showMonth.setText("" + month);

}

测试用例：4

Year类：新建年月的标签并设置“上年”和“下年”的属性

1. getYear()：

public int getYear() {

return year;

}

测试用例：2019

1. setYear（）：

public void setYear(int year) {

this.year = year;

showYear.setText("" + year);

}

测试用例：2018

Notepad类：实现对记事本面板的属性设置

1. setYear（）：

public void setYear(int year) {

this.year = year;

}

测试用例：2019

1. setMonth（）：

public void setMonth(int month) {

this.month = month;

}

测试用例： 5

1. setDay（）：

public void setDay(int day) {

this.day = day;

}

测试用例：1

4. setText（）：

public void setText(String s) {

text.setText(s);

}

测试用例：ss

5.setIFline（）：

public void setIFLine(int year, int month, int day) {

IFLine.setText("" + year + "年" + month + "月" + day + "日");

测试用例：2019年5月1日

6.lookCalender（）：

public void lookCalendar(int year, int month, int day) {

String key = "" + year + "" + month + "" + day;

try {

FileInputStream inOne = new FileInputStream(file);

ObjectInputStream inTwo = new ObjectInputStream(inOne);

table = (Hashtable) inTwo.readObject();

inOne.close();

inTwo.close();

}

测试用例： 无记录 空记录

“123” 有记录，但是需要手动操作

测试结果：

对于类的覆盖100%。

对于语句的覆盖 79%。

对于白盒测试的反思与学习：

由于我们不太明白鼠标点击等事件的白盒测试方法，因此对于语句的覆盖程度略低。

# 建议措施

[根据对测试结果和主要测试评测结果所进行的评估，建议任何可取的措施。]

# 图

|  |  |
| --- | --- |
| 测试编号 | 测试结果（图） |
| 1 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\LQMT~YUO90%MSC7VHS(T[ON.png |
| 2 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\C@U_7RGE@W)W%{FTQ@4BE@I.png |
| 3 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\3Q]QZ68EM1D{ZMGB(9QQQSO.png |
| 4 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\4WJXF`~T$}FUG44)NGG8PRB.png |
| 5 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\OXC_Y%0N)DPD)6F`KO)S]LY.png |
| 6 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\_8HX{DR}I]BD]C)_7~@}$YF.png |
| 7 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\AIB452R2{QOEN~BQ5918}AP.png |
| 8 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\3%W$(PR0MGR4M(C}@%I`WFI.png |
| 9 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\IHE~SY~XZIDF}IHNA$4DK%3.png |
| 10 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\~I4_$ZMF9(4(631Y2PBT__U.png |
| 13 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\90$`GU{24OZ@P@)~$FR%A[V.png |
| 11 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\VF`S7L[FGHCMBV@EEG~)WQH.png |
| 14 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\F{TC%S6X9}]X71$DD$SMSL8.png |
| 12 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\[HX[K9DUSBUS8FU@9X_O87U.png |
| 15 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\~(N39EG{8T`(F_FE9~WD~W2.png |
| 16 | 无 |
| 17 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\IUC5]KVTR}VXG%KHAFCNE11.png |
| 18 | C:\Users\survivor\Documents\Tencent Files\674041200\Image\Group\E{LU(4_${YPF31ZHHV[[Q5H.png |
| 19 |  |
| 20 |  |
| 21 |  |
| 22 |  |
| 23 |  |
| 24 |  |