國立臺北科技大學

資訊工程系

軟體測試與驗證

Test Result Summary

Turbo Editor

Team#6

專案成員：古兆瑋（108598019）

陳冠穎（108598034）

劉孝忠（108598044）

民國 109 年 6 月 23 日

Table of Contents

Change History

Version	Primary Author(s)	Description of Version	Date Completed
1.0.0	古兆瑋、陳冠穎、劉孝忠	樣式及內容初稿。	2020/06/23

1. Test Plan and Result

1.1 Description

本次測試將會以 SUT 中，以 Turbo Editor 的 Feature List 中，較常使用到的主要項目作為 Test Items，並將其歸類為相對應的 Use Cases，使往後 Test Cases 設計時有相對應的 Use Cases。

1.2 Features To Be Tested

Feature ID	Description	Use Case
F-T-01	Create New File	Create File
F-T-02	Open File	Read File
F-T-03	Read File's Info	
F-T-04	Rename File	Update File
F-T-05	Edit File	
F-T-06	Undo Editing	
F-T-07	Redo Editing	
F-T-08	Search File with REGEX	Search File
F-T-09	Search File with MATCH CASE	
F-T-10	Search File with REPLACE	
F-T-11	Set Line Numbers	Set Configuration
F-T-12	Set Syntax Highlight	
F-T-13	Set Wrap Content	
F-T-14	Set Use Monospace	
F-T-15	Set Read Only	
F-T-16	Set Font Size	
F-T-17	Set Theme	
F-T-18	Set Accessory View	
F-T-19	Set Use the storage Access Framework	
F-T-20	Set Keyboard Suggestion and Swipe	
F-T-21	Set Auto Save	
F-T-22	Set Encoding	
F-T-23	Set Ignore Back Button	
F-T-24	Set Split The Text If Too Long	
F-T-25	Set Fullscreen Mode	
F-T-26	Set Open Last Viewed File At Startup	

1.3 Features Not To Be Tested

Feature ID	Description	Use Case
F-NT-01	Create New Folder	Create File
F-NT-02	Open Folder	Read File
F-NT-03	Share File	
F-NT-04	Rename Folder	Update File
F-NT-05	Edit Folder	

1.4 Item Pass/Fail Criteria

Test Item 的 Pass/Fail 判斷標準，分為下列兩種方式：

一、以 User Scenario Technique 設計之測試案例：以一功能之敘述句，在測試執行後是否成立為 Pass/Fail 標準，假使功能測試執行後，使敘述句成立則 Pass；反之則 Fail，而本次測試之目標為使所有與 Use Case 有關以及與 1.2 Features To Be Tested 有關之 Test Case 通過測試(Pass)，證明軟體有達成使用案例的需求。

二、以 Control-Flow Graph Coverage Technique 設計之測試案例：以 Basis Path Approach，須覆蓋所有分支，意即 Statement Coverage 及 Branch Coverage 是 100%，證明測試案例是有實際將該段程式碼執行過的。

不論判斷標準為何，最終希望本次測試能夠達成 80%的 Code Coverage，證明因為測試涵蓋這麼高的覆蓋率，所以測試的結果是可以信任的。

1.5 Test Approach

本次計畫使用 User Scenarios 測試技巧，將使用者操作案例之步驟逐步以 APP UI 測試自動化框架之腳本撰寫之，進行自動化測試。

1.6 Test Design Identification

Test ID	Description	Related Use Case
TD-01	Create New File	Create File
TD-02	Open File	Read File
TD-03	Read File’s Info	
TD-04	Rename File	Update File

TD-05	Edit File	
TD-06	Undo Editing	
TD-07	Redo Editing	
TD-08	Search File with REGEX	Search File
TD-09	Search File with MATCH CASE	
TD-10	Search File with REPLACE	
TD-11	Set Line Numbers	Set Configuration
TD-12	Set Syntax Highlight	
TD-13	Set Wrap Content	
TD-14	Set Use Monospace	
TD-15	Set Read Only	
TD-16	Set Font Size	
TD-17	Set Theme	
TD-18	Set Accessory View	
TD-19	Set Use the storage Access Framework	
TD-20	Set Keyboard Suggestion and Swipe	
TD-21	Set Auto Save	
TD-22	Set Encoding	
TD-23	Set Ignore Back Button	
TD-24	Set Split The Text If Too Long	
TD-25	Set Fullscreen Mode	
TD-26	Set Open Last Viewed File At Startup	

1.7 Test Case Identification

Test Case ID	Description	Related Test Item
TC-01	Create File	TD-01
TC-02	Read File	TD-02、TD-03
TC-03	Update File	TD-04、TD-05、TD-06、TD-07
TC-04	Search File	TD-08、TD-09、TD-10
TC-05	Set Configuration	TD-11、TD-12、TD-13、TD-14、TD-15、TD-16、TD-17、TD-18、TD-19、TD-20、TD-21、TD-22、TD-23、TD-24、TD-25、TD-26
TC-06	Back When Create File	TD-01
TC-07	Close APP When Create File	TD-01

TC-08	undo	
TC-09	replaceTextKeepCursor	

1.8 Test Result

app-pro

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
shared.turboeditor.home		50%		40%	330	501	485	1,130	70	182	15	34
shared.turboeditor.util		51%		24%	174	218	280	407	68	102	7	20
shared.turboeditor.dialogfragment		33%		14%	102	131	255	394	54	81	18	30
shared.turboeditor.explorer		0%		0%	76	76	199	199	25	25	5	5
shared.turboeditor.home.texteditor		43%		32%	75	105	132	227	32	53	3	12
shared.turboeditor.adapter		29%		13%	39	48	107	145	18	25	6	10
shared.turboeditor.util.systemui		0%		0%	61	61	128	128	37	37	8	8
shared.turboeditor.views		41%		18%	42	59	55	111	19	34	1	5
shared.turboeditor.preferences		79%		45%	31	97	58	211	23	87	0	20
shared.turboeditor.files		57%		30%	25	36	42	93	3	13	1	4
shared.turboeditor.markdown		0%		n/a	2	2	7	7	2	2	1	1
shared.turboeditor.application		41%		0%	1	3	5	10	0	2	0	1
com.maskyn.fileeditorpro		60%		n/a	1	2	1	2	1	2	0	1
Total	9,139 of 16,570	44%	973 of 1,373	29%	959	1,339	1,754	3,064	352	645	65	151

Test Case ID	Description	Pass / Fail
TC-01	Create File	Pass
TC-02	Read File	Pass
TC-03	Update File	Pass
TC-04	Search File	Pass
TC-05	Set Configuration	Pass
TC-06	Back When Create File	Pass
TC-07	Close APP When Create File	Pass
TC-08	undo	Pass
TC-09	replaceTextKeepCursor	Pass
Pass Rate		9/9 (100%)
Code Coverage		44% (<80%)

1.9 Test Analysis

依據 1.4 Item Pass/Fail Criteria 之敘述，以 User Scenarios 方法設計之測試案例編號 TC-01 至 TC-05 有確實以相關 Use Case 進行設計；特意模擬使用者可能會造成的例外狀況之測試案例編號 TC-06 至 TC-07 有確實以相關 Use Case 進行設計；以 Control-Flow Graph 方法設計之測試案例編號 TC-08 及 TC-09 有確實以控制流程圖進行設計，也確實按照設計去實作，並執行測試通過。但程式碼覆蓋率(Code Coverage)未達到預期的 80%，主要原因是我們高估了 User Scenarios 黑箱測試可以達到的程式碼覆蓋率，我們以為以 APP 最主要且幾乎是所有的功能去測試的話，應該可以達到至少 80%的期望，但可能是 Use Case 的

Extension 寫得不夠多，應該要多考慮些例外，才能將程式碼覆蓋率提升，亦或是以白箱測試的方式去提升。

1.10 Software Bug Found through Testing

```
if (Device.hasKitKatApi() && TextUtils.isEmpty(viewModel!!.greatUri!!.filePath)) {
    var newUri: Uri? = null
    try {
        DocumentsContract.renameDocument(contentResolver, viewModel!!.greatUri!!.uri!!, result)
    } catch (e: FileNotFoundException) {
        newUri = null
    }
```

圖 1.10.1: 原始程式碼

```
override fun onEdittextDialogEnded(result: String, hint: String, action: EditTextDialog.Actions) {

    if (Device.hasKitKatApi() && TextUtils.isEmpty(viewModel!!.greatUri!!.filePath)) {
        var newUri: Uri? = null
        newUri = try {
            DocumentsContract.renameDocument(contentResolver, viewModel!!.greatUri!!.uri!!, result)
        } catch (e: FileNotFoundException) {
            null
        }
```

圖 1.10.2: 修訂後之程式碼

在測試的過程中，我們發現此應用程式 rename 檔案或資料夾的功能沒有作用，所以在實作測試案例時實際去追蹤了一下程式碼，才發現原來變數 newUri 從頭到尾都沒有被賦予值，後來重新理解了原先的程式碼價購後，我們重新將 try-catch 的區塊賦予給變數 newUri 後，成功的解決無法 rename 檔案或資料夾的問題。

1.11 Summary

總結來說，本次測試根據了 Software Test Plan 的目標訂定測試方向以及分析測試需求，並且在 Test Design Specification 依據 Software Test Plan 的規格去分析 Test Item，之後於 Test Case Specification 進行對各個 Test Item 有關之測試案例的設計，而最後在執行已實作測試案例之後，於本文件分析本次軟體測試的結果與紀錄我們對於本次軟體測試的總結，最終雖然程式碼覆蓋率並沒辦法達成需求，也並未以所有於課堂上教過的 Approach 進行設計，但我們還是有學習到如何以 APP 的測試工具進行測試的設計，以及使用部分 Approach 設計測試案例的能力。

Glossary

SUT

軟體系統測試則通常是以被測系統（System Under Test，SUT）表示正在被測試的系統，目的是測試系統是否能正確操作。這一詞語常用於軟體測試中。[1]

Android

Android（讀音：英：['ændrɔɪd]，美：[ˈæn¸drɔɪd]），中文常譯作安卓或安致，是一個基於 Linux 核心的開放原始碼行動作業系統，由 Google（Google）成立的開放手機聯盟持續領導與開發，主要設計用於觸控螢幕行動裝置如智慧型手機和平板電腦與其他可攜式裝置。[2]

APP

A short of “Application”

Turbo Editor

Simple, Powerful and Open Source Text Editor for Android. Built following the latest Android Guidelines to make the experience the most intuitive on the Store, also on Tablets![3][4]

Espresso

The core API is small, predictable, and easy to learn and yet remains open for customization. Espresso tests state expectations, interactions, and assertions clearly without the distraction of boilerplate content, custom infrastructure, or messy implementation details getting in the way.

Espresso tests run optimally fast! It lets you leave your waits, syncs, sleeps, and polls behind while it manipulates and asserts on the application UI when it is at rest.[5]

UI Automator

The UI Automator testing framework provides a set of APIs to build UI tests that perform interactions on user apps and system apps. The UI Automator APIs allows you to perform operations such as opening the Settings menu or the app launcher in a test device. The UI Automator testing framework is well-suited for writing black box-style automated tests, where the test code does not rely on internal implementation details of the target app.[6]

Reference

[1] 被測系統(Wikipedia)
https://zh.wikipedia.org/wiki/%E8%A2%AB%E6%B5%8B%E7%B3%BB%E7%BB%9F
[2] Android(Wikipedia) https://zh.wikipedia.org/wiki/Android
[3] Turbo Editor(Google Play)
https://play.google.com/store/apps/details?id=com.maskyn.fileeditor
[4] Turbo Editor(Github) https://github.com/vmihalachi/turbo-editor
[5] Espresso https://developer.android.com/training/testing/espresso
[6] UI Automator https://developer.android.com/training/testing/ui-automator
« »