Software Test Plan

for

Iris: Email Reader Application

Version 1.00

Scott Arnette

Joseph LaCava

Derek Ouzia

Bryan Smith

University of Virginia’s College at Wise

Department of Mathematics and Computer Science

December 6th 2015

Table of Contents

Table of Contents ii

Revision History iii

1. Introduction 1

1.1 Objectives 1

1.2 Testing Strategy 1

1.3 Scope 2

1.4 Reference Material 2

1.5 Definitions and Acronyms 3

2. Test Items 3

2.1 Program Modules 3

2.2 User Procedures 4

2.3 Operator Procedures 4

3. Features 4

3.1 Features to be Tested 4

3.2 Features Not to be Tested 4

4. Approach 4

4.1 Component Testing 4

4.2 Integration Testing 5

4.3 Interface Testing 5

4.4 Regression Testing 5

4.5 Acceptance Testing 5

4.6 Beta Testing 5

5. Pass/Fail Criteria 6

5.1 Suspension Criteria 6

5.2 Resumption Criteria 6

5.3 Approval Criteria 6

6. Testing Process 6

6.1 Test Deliverables 6

6.2 Testing Tasks 6

6.3 Responsibilities 7

6.4 Resources 7

6.5 Schedule 7

7. Environmental Requirements 7

7.1 Hardware 7

7.2 Software 7

7.3 Security 7

7.4 Tools 8

7.5 Publications 8

7.6 Risks and Assumptions 8

8. Change Management Procedures 8

9. Test Procedures 8

10. Requirements Matrix 9

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Bryan Smith | 12/6/15 | Initial Creation | 1.00 |
|  |  |  |  |

# Introduction

This Software Test Plan (STP) will provide an overview of testing procedures and information for the Email Reader Application, also known as Iris. It will be referred to as Iris, or simply, the application, henceforth. The plan will identify items to be tested, the features to be tested, and the types of testing to be performed.

Primary focus of the testing of the application will be making sure it is backwards compatible (works on older versions) with older versions of Android back to API 11 (Honeycomb).

## Objectives

* Identify existing project information and the software that should be tested.
* List the recommended test requirements (high level).
* Recommend and describe the testing strategies to be employed.
* Identify the required resources and provide an estimate of the test efforts.
* List the deliverable elements of the test activities.

## Testing Strategy

The Android framework includes an integrated testing framework that helps test all aspects of the application and the SDK tools include tools for setting up and running test applications. Whether you are working in Eclipse with ADT, Android Studio, or working from the command line, the SDK tools help you set up and run tests within an emulator or the physical devices being targeted.

The testing framework has these key features:

* Android test suites are based on JUnit. Plain Junit can be used as well to test a class that does not call the Android API, or Android's JUnit extensions to test Android components.
* The Android JUnit extensions provide component-specific test case classes. These classes provide helper methods for creating mock objects and methods that help control the lifecycle of a component.
* Test suites are contained in test packages that are similar to main application packages, so you don't need to learn a new set of tools or techniques for designing and building tests.
* The SDK tools for building and tests are available in Eclipse with ADT, Android Studio, and also in command-line form for use with other IDEs. These tools get information from the project of the application under test and use this information to automatically create the build files, manifest file, and directory structure for the test package.
* The SDK also provides monkeyrunner, an API for testing devices with Python programs, and UI/Application Exerciser Monkey, a command-line tool for stress-testing UIs by sending pseudo-random events to a device.

A large subset of testing for Iris will be White Box testing and Integration tests.

Refer to section 5 of this document for a detailed list of specific test plans.

## Scope

Scheduled and unscheduled changes will be managed via Issues and Pull requests integrated in BitBucket. BitBucket is the Git remote hosting service used for the Iris project. Git is a source control tool.

Updates to already released and distributed versions of Iris will be performed via the Google Play Store and will show up automatically if the user has Iris installed. A delayed or limited rollout can be defined in Google Play so that only small subsets of users see the update initially. This will ensure that if there was something that went unnoticed and made it in the release, we will receive feedback from this small subset of users and limit the impact on both user frustrations and/or network infrastructure for distribution

Lastly, Google Play Store may not be used in the event that licensing does not want to be paid in order to use Google Play Store. In this situation, the compiled APK can be distributed by common methods, including but not limited to: email, website download, content management systems, or manual command line installation using ADB.

## Reference Material

* Iris: Email Reader Application – Software Requirements Specification
* Iris: Email Reader Application – Software Design Document
* Android Testing <http://developer.android.com/tools/testing/index.html>

## Definitions and Acronyms

* ADB – Android Debug Bridge. Versatile command line tool that lets you communicate with an emulator instance or connected Android-powered device
* APK - Android application package is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware.
* API – Application Programming Interface. A set of routines, protocols, and tools that govern a software specification.
* GUI – Graphical User Interface. An interface that allows users to interact with electronic devices through icons and visual indicators.
* Iris – The name of the Email Reader Application. The project that this STP is for.
* OS – Operating System. The software that the product runs on.
* REST API - Representational state transfer application programming interface. A light weight web based API. The client does not need to know the structure of the API but the server provides the information the client needs to interface with the service.
* SDD – Software Design Document
* SRS – Software Requirements Specification.
* STP – Software Test Plan

# Test Items

## Program Modules

<Outline testing to be performed by the developer for each module being built.>

**TODO: Next Semester**

## User Procedures

Due to the small size of the Iris project, any user documentation will be re-read by the team and manually edited to reflect the changed or correct information.

## Operator Procedures

The application shall check for environment compatibility upon startup. Thus it is self-checking and no procedures are needed other than initial installation of the application. The only requirement for installation is having an Android Device with API 11+.

# Features

## Features to be Tested

<Identify all software features and combinations of software features to be tested. Identify the test design specifications associated with each feature and each combination of features.>

**TODO: Next Semester**

## Features Not to be Tested

<Identify all features and specific combinations of features that will not be tested along with the reasons.>

**TODO: Next Semester**

# Approach

<Describe the overall approaches to testing. The approach should be described in sufficient detail to permit identification of the major testing tasks and estimation of the time required to do each task. Identify the types of testing to be performed along with the methods and criteria to be used in performing test activities. Describe the specific methods and procedures for each type of testing. Define the detailed criteria for evaluating the test results.>

<For each level of testing there should be a test plan and the appropriate set of deliverables. Identify the inputs required for each type of test. Specify the source of the input. Also, identify the outputs from each type of testing and specify the purpose and format for each test output. Specify the minimum degree of comprehensiveness desired. Identify the techniques that will be used to judge the comprehensiveness of the testing effort. Specify any additional completion criteria (e.g., error frequency). The techniques to be used to trace requirements should also be specified.>

**TODO: Next Semester**

## Component Testing

<Testing conducted to verify the implementation of the design for one software element (e.g., unit, module) or a collection of software elements. Sometimes this is called unit testing. The purpose of component testing is to ensure that the program logic is complete and correct and ensuring that the component works as designed.>

**TODO: Next Semester**

## Integration Testing

<Testing conducted in which software elements, hardware elements, or both are combined and tested until the entire system has been integrated. The purpose of integration testing is to ensure that design objectives are met and ensures that the software, as a complete entity, complies with operational requirements. Integration testing is also called System Testing.>

**TODO: Next Semester**

## Interface Testing

It has been determined that due to the limited GUI elements in the Iris project, that most UI elements will be tested manually by the team. It would be counterproductive to spend the time to setup the UI testing platform Google has for Android, as it would take longer and likely be less accurate due to the nature of the Iris project. The UI testing platform may be utilized to verify GUI states (does it save its state when it is moved from background to foreground and visa-versa?) however.

## Regression Testing

<Testing done to ensure that that applied changes to the application have not adversely affected previously tested functionality.>

**TODO: Next Semester**

## Acceptance Testing

<Testing conducted to determine whether or not a system satisfies the acceptance criteria and to enable the customer to determine whether or not to accept the system. Acceptance testing ensures that customer requirements' objectives are met and that all components are correctly included in a customer package.>

**TODO: Next Semester**

## Beta Testing

<Testing, done by the customer, using a pre-release version of the product to verify and validate that the system meets business functional requirements. The purpose of beta testing is to detect application faults, failures, and defects.>

**TODO: Next Semester**

# Pass/Fail Criteria

<Specify the criteria to be used to determine whether each item has passed or failed testing.>

**TODO: Next Semester**

## Suspension Criteria

<Specify the criteria used to suspend all or a portion of the testing activity on test items associated with the plan.>

**TODO: Next Semester**

## Resumption Criteria

<Specify the conditions that need to be met to resume testing activities after suspension. Specify the test items that must be repeated when testing is resumed.>

**TODO: Next Semester**

## Approval Criteria

<Specify the conditions that need to be met to approve test results. Define the formal testing approval process.>

**TODO: Next Semester**

# Testing Process

<Identify the methods and criteria used in performing test activities. Define the specific methods and procedures for each type of test. Define the detailed criteria for evaluating test results.>

**TODO: Next Semester**

## Test Deliverables

<Identify the deliverable documents from the test process. Test input and output data should be identified as deliverables. Testing report logs, test incident reports, test summary reports, and metrics' reports must be considered testing deliverables.>

**TODO: Next Semester**

## Testing Tasks

<Identify the set of tasks necessary to prepare for and perform testing activities. Identify all intertask dependencies and any specific skills required.>

**TODO: Next Semester**

## Responsibilities

<Identify the groups responsible for managing, designing, preparing, executing, witnessing, checking, and resolving test activities. These groups may include the developers, testers, operations staff, technical support staff, data administration staff, and the user staff.>

**TODO: Next Semester**

## Resources

<Identify the resources allocated for the performance of testing tasks. Identify the organizational elements or individuals responsible for performing testing activities. Assign specific responsibilities. Specify resources by category. If automated tools are to be used in testing, specify the source of the tools, availability, and the usage requirements.>

**TODO: Next Semester**

## Schedule

<Identify the high level schedule for each testing task. Establish specific milestones for initiating and completing each type of test activity, for the development of a comprehensive plan, for the receipt of each test input, and for the delivery of test output. Estimate the time required to do each test activity.>

<When planning and scheduling testing activities, it must be recognized that the testing process is iterative based on the testing task dependencies.>

**TODO: Next Semester**

# Environmental Requirements

<Specify both the necessary and desired properties of the test environment including the physical characteristics, communications, mode of usage, and testing supplies. Also provide the levels of security required to perform test activities. Identify special test tools needed and other testing needs (space, machine time, and stationary supplies. Identify the source of all needs that is not currently available to the test group.>

## Hardware

* An Android device (phone or tablet) is needed for testing the application.
* The device must have touch support or mouse and keyboard support.
* The device must have connectivity to a network connection
* The network connection shall be speeds similar to that of Mobile 3G or higher (144Kb/s+).
* Testing can be performed on an emulated device so long as it meets the above specifications as well as the software specifications below.

## Software

* The device must be running Android OS 3 (API 11) or higher
* The device must have Google Play Services v8.3 or higher installed

## Security

<Identify the testing environment security and asset protection requirements.>

**TODO: Next Semester**

## Tools

<Identify the special software tools, techniques, and methodologies employed in the testing efforts. The purpose and use of each tool shall be described. Plans for the acquisition, training, support, and qualification for each tool or technique.>

**TODO: Next Semester**

## Publications

<Identify the documents and publications that are required to support testing activities.>

**TODO: Next Semester**

## Risks and Assumptions

<Identify significant constraints on testing such as test item availability, test resource availability, and time constraints. Identify the risks and assumptions associated with testing tasks including schedule, resources, approach and documentation. Specify a contingency plan for each risk factor.>

**TODO: Next Semester**

# Change Management Procedures

<Identify the software test plan change management process. Define the change initiation, change review, and change authorization process.>

**TODO: Next Semester**

# Test Procedures

<Fill out the following table for each test procedure.>

**TODO: Next Semester**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Procedure Number:** | |  | | |
| **Date Tested:** | |  | | |
| **Test Performed By:** | |  | | |
| **Project Name:** | |  | | |
| **Software Version:** | |  | | |
| **Related Requirements:** | |  | | |
| # | Test Step Description | | Expected Result | Passed |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |

# Requirements Matrix

*<Provide a cross reference that traces the test procedures to the requirements in your SRS document. Use a tabular format to show which tests satisfy each of the functional requirements from the SRS. Refer to the functional requirements by the numbers/codes that you gave them in the SRS.>*

**TODO: Next Semester**