Software Requirements Specification

for

WinterHasCome(Weatherapp)

Version 1.0 approved

Prepared by Prashanth Murali

Arizona State University

12/01/2016

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This document is meant to describe the features of WinterHasCome(weatherapp). So as to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other. The weatherapp gathers weather data of the required location from the openweatherapp api and displays the details on the screen. The weather details are obtained as a JSON file.

1.2 Product Scope

The weatherapp is used to provide weather data such as temperature, sunrise time, sunset time, wind direction, current conditions for the location of the user's choice. The app allows the User to enter the location i.e town or city for which they require the weather data. The app is designed to perform the data collection operations in the background method which reduces the load on the target android device.

1.3 Definitions

SRS- Software Requirement Specification
Target Device- Device on which the app is to be run
Weatherapp- the name I will be using to refer to the app also Known as WinterHasCome

1.3.1 Overview

Users can use the weatherapp to obtain easy access to everyday weather at a number of locations across the globe.

1.4 Additional Information

This SRS document is intended to give a brief depiction of the product's functionality, the hardware requirements, software requirements, functional and non-functional requirements for implementing the product. The forthcoming sections of the SRS provide a detailed description of the project, specify the requirements and explain the various scenarios of operation.

2. Overall Description

The weatherapp obtains weather data using openweatherapi from the servers of Openweather.

2.1 Product Functions

The weatherapp contains data such as the URL of the JSON file to be parsed and the URL of the image to be downloaded preloaded into it at the start. The ExtractWeatherData class obtains the individual JSONObjects and extracts data from each class and stores it. Then the MainActivity class obtains the data in the respective fields and they are displayed in their appropriate interface elements.

2.2 Operating Environment

The weatherapp will operate in android versions 6.0.1 and higher. It cannot be installed in windows or ios or any other operating system. The target device must be connected to the internet.

2.3 User Documentation

A read me document is provided for a first time user to use the project.

2.4 Assumptions and Dependencies

Users of the product are required to have basic knowledge about the usage of android phones, must be able to spell in English language and must have android 6.0.1 or higher installed on their mobile device. Users are required to use a mobile phone as the app is incompatible with tablets or any other android devices.

3. Functional Requirements

- **Android Device with Android V 6.0.1 or higher**

  The app requires the user to operate an android device with android V 6.0.1 or higher in order to function.

- **OpenweatherMap Api**

  The app requires the use of a weather api such as Openweatherapi to obtain the weather details from and parse through.

- **Access to the Internet**

  The weatherapp cannot function if the device is not connected to the internet.

- ***Display***

The app must display all the weather information obtained from the each JSONObject appropriately.

- ***Change Location***

The app must include a button to enable the user to request weather data for a location of their choice.

3.2 Technical Issues

The weatherapp works on a client server architecture where the user's device acts as the client and the Openweatherapp api works as the server and handles user requests.

4. Non-functional Requirements

4.1 Performance Requirements

The product is expected to perform optimally with no errors or significant reduction in performance at any load of operation. The onbackground method of the Asynctask ensures that the major operations are performed in the background.

4.2 GUI Requirement

The GUI consists of several TextViews to display the various weather condition details. An ImageView is used to download the weather icons and display them along with the other details. The user can provide input and search for weather details of the city they desire.

4.3 Security Requirements

The app does not have any security requirements as it does not handle any safety critical data. It obtains the data from the Openweatherapp servers and displays it on the screen.

4.4 Reliability

The weatherapp consumes very little memory and hence has very little toll on the internal RAM of the user's android device. Thus even when using a large amount of memory and under stress, the app would be able to run as it consumes minimal amount of RAM(20MB). All the weather data is obtained from the Openweathermap database which is exceptionally reliable. All coding standards and Javadoc documentation standards are followed in developing the app which makes it easily maintainable. The app can be converted to previous sdk versions with relative ease which makes it very portable. The functioning of the app is very trivial and the user interface is simple and easy to understand.

4.5 Maintainability

All coding standards and Javadoc documentation standards are followed in developing the app which makes it easily maintainable.

4.6 Portability

The app can be converted to previous sdk versions with relative ease which makes it very portable.

4.7 Availability

All the weather data is obtained from the Openweathermap database which is exceptionally reliable and has an average downtime of 0.2% hours per week.

5. Operational Scenario

The App displays the weather conditions such as Temperature, wind, current conditions, sunrise and sunset times. Users can search for the weather conditions in the city of their choice by clicking on the change city menu button which opens up a dialog box for the user to enter input.