Software Engineering Group 09 Project
Maintenance Manual (Desktop)

Author: Runar Reve [rur7]
Config Ref: SE.GP09.MAN_DESKTOP
Date: 8th May 2019
Version: 1.0
Status: Release

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB

CONTENTS

1. INTRODUCTION

1.1 Purpose of this Document

This document is to provide information and answers to specific questions regarding the maintenance of the desktop application developed for our project.

1.2 Scope

This aims to outline any potential pitfalls or problems that could be encountered when maintaining/updating the software, and outlines areas that could most use improving or updating.

1.3 Objectives

The objective of this document is to provide answers, warnings and general information to be used by programmers attempting to maintain the android application.

2. PROGRAM DESCRIPTION

The program is an application we have created to allows a user to view, create, and delete different list of pubs. For each list the user can create, edited, view and delete pubs with various details from each list Each pub will have a name, a set number of characteristics, a description, location, and a way to store images. The program will store all the information in a PostgreSQL server and can be used by an android device to view the pubs, filter them, and create different types of tours with them.

3. PROGRAM STRUCTURE

The programs structure is described in the *Design Specification* [3]. But to summarise the design: The structure is based on three components. *PubEngin*: which is the main brain of the program and will do the computing and connecting everything together. *DbCommunicator* which will communicate with our PostgreSQL server to send or receive the information of the system. And the *User Interface*, which is where the user will interact with the program and use it functionalities.

4. ALGORITHMS

There are not really any complicated algorithms in this application, except for how the *DbCommunicator* communicates which the database. When receiving information from the database, the functions will extract the information and create the wanted object, Town or Pub. When sending information, the object will be deconstructed into a string that can be used as a SQL command that the server can understand and execute the command.

To convert the address and postcode, there is an algorithm to connect to an URL. It will return a Json file from the connected URL and from that file it will deconstruct it to extract the coordinates of the address.

To view more information about these algorithms, see the *Design Specification [3]*.

5. MAIN DATA AREAS

Objects:

Town:

An ArrayList of towns will be extracted from the database each time the application gets started up. And will be used to store the lists available for the user to select from.

Pub:

An ArrayList of pub objects will store all the information locally that is from the list. Each time a change is made to one of the pubs, it will also be updated in the database. When connecting to a new town it will re pull the list of pubs of that town.

To view more information about the main data areas, see the *Design Specification [3]*.

6. FILES

Our application does not create or read any physical files, and instead connects to a database stored on the Aberystwyth university server and reads information from there.

7. INTERFACES

Our application has the *DbReceiverInterface* and the *DbUpdateInterface* interfaces. These are not reused internally in this application, but the *DbReceiverInterface* is reused in the mobile application to communicate with the database, because there are differences in how an android app and a regular java app communicate with a database. There are also possibilities to expand using these interfaces to other systems

8. SUGGESTIONS FOR IMPROVEMNETS

The easiest and best improvement to do is to add fields for the user to enter/modify the raw coordinates of the exact location of the pub to make the application better fit the requirements in the *Requirement Specification [4]*.

Implement a more user-friendly method of storing images and store them in the database to make the application better fit the requirements in the *Requirement Specification [4]*.

Edit so that the default characteristics is set to no when the user is selecting characteristics, compared to the user having to select each characteristic before being able to save them. This will make the design more user-friendly.

Add more automatic Junit test to more easily the system to make better sure the system is functional after modification and updates.

Add a google maps popup so that the user can select the location using an UI instead of entering coordinates or addresses. This would make the application better fit the *User Interface Specification [5]*

Add a popup for the user to view, add, or delete different images to make the application better fit the *User Interface Specification [5]*

Add/improve comments of the code to make the it more readable and more up to the *Java Coding Standards [6]*

Add missing authors to the code to make it more up to the *Java Coding Standards [6]*

9. THINGS TO WATCH FOR WHEN MAKING CHANGES

Due to the structure of the program, the future developer of this application will need to remember to pass all the necessary objects and variables through buttons, most notably: current list town and current list of pubs.

For details on how to improve this see section 8 of this document.

10. PHYSICAL LIMITATIONS OF THE PROGRAM

Recommended java version:1.8 compiler (might work with newer versions)

Recommended disc space 5 MB (needs 2MB)

Recommended RAM >400 MB (For data from the database)

Needs to be connected to the Aberystwyth University's network

11. REBUILDING AND TESTING

If rebuilding the system and you get encounter by errors when first building the app, follow the following list

- In case of compiling due to missing directories/libraries/settings errors:
- Make sure that all the needed libraries are included, as of original release: json-20180130.jar, junit-4.12.jar, and postgresql-42.2.5.jar. All these will be released under the repository \lib
- Make sure that the project \src directory is set to src directory
- Make sure that the project \out directory is set to out directory
- Make sure that the project can locate the main function located at: GroupProject09\src\desktop_software\src\uk\ac\aber\cs221\group09*PubEngin.java*
- If there are any other queries, the easiest solution is often to locate the .idea in the GroupProject09\src\desktop_software directory and replace this with the .idea under: C:\Users\rever\Desktop\GroupProject09\dev\post20190403\IDEA backup (.idea is an ignored directory by *.gitignored* in the \src folder due to frequent development clashes)

The few automatic testes are in the system has are located under:
GroupProject09\src\desktop_software\src\uk\ac\aber\cs221\group09\DbTest

REFERENCES

[1] Software Engineering Group Projects: General Documentation Standards. C. J. Price, N. W. Hardy, B.P. Tiddeman. SE.QA.03. 1.8 Release
[2] Software Engineering Group Projects – Producing a Final Report. C. J. Price. SE.QA.10. 2.2 Release
[3] Software Engineering Group 09 Projects – Design Specification. R. Reve SE.GP09.DESIGNSPEC. 1.1 Release
[4] Software Engineering Group Projects: Requirements Specification. C. J. Price. SE.QA.CSRS. 1.1 Release
[5] Software Engineering Group 09 Projects: User Interface Specification. B. Weatherley, B. James. SE.GP09.UISPEC. 1.3 Release
[5] Software Engineering Group Projects: Java Coding Standards. C. J. Price, A. McManus. SE.QA.09. 2.0 Release

DOCUMENT HISTORY

Version	*CCF No.*	*Date*	*Changes made to document*	*Changed by*
1,0	N/A	08/05/2019	Original Version	RUR7