Group 12 Software Engineering Group Project

Maintenance Manual Documentation

Author:	E Kalisiak, S MW Twerdochlib & J Kus
Config. Ref.:	SE.12.MD
Date:	7th May 2019
Version:	1
Status:	Release

Department of Computer Science,
Aberystwyth University,
Aberystwyth,
Ceredigion, SY23 3DB,
U.K.

CONTENTS

1 INTRODUCTION 2
1.1 Purpose of this document. 2
1.2 Scope 2
1.3 Objectives 2

2 DESKTOP 2
2.1 Program Description 2
2.2 Program Structure 2
2.2.1 See [1] section 2.2.1 and 2.2.3 for a list of classes with a description. 2
2.3 Algorithms 3
2.4 The Main Data Area 3
2.5 Files 3
2.6 Interfaces 3
2.7 Suggestions For Improvements 3
2.8 Things Wo Watch When Making Changes 3
2.9 Physical Limitations Of The Program 3
2.10 Rebuilding And Testing 4

3 MOBILE 4
3.1 Program Description 4
3.2 Program Structure 4
3.2.1 List of modules. 4
3.2.2 List of methods 4
3.3 Algorithms 4
3.4 The Main Data Area 5
3.5 Files 5
3.6 Interfaces 5
3.7 Suggestions For Improvements 5
3.8 Things To Watch For When Making Changes 5
3.9 Physical Limitations Of The Program 5
3.10 Rebuilding And Testing 6

REFERENCES 6

DOCUMENT HISTORY 6

1 INTRODUCTION

1.1 Purpose of this document.

The goal of this document is to answer all of the specific questions that installers or maintainers of the software can have.

1.2 Scope

The project maintenance manual document describes the program, its structure and algorithms used in it. It includes an information about interfaces, suggestions for improvements and things to watch when making changes. It mentions physical limitations of the program, a way of rebuilding and testing the software.

1.3 Objectives

The objectives of this document are:

- Program description.
- Program structure.
- Algorithms.
- The main data area.
- Files.
- Interfaces.
- Suggestions for improvement.
- Things to watch for when making changes.
- Physical limitations of the program.
- Rebuilding and testing.

2 DESKTOP

2.1 Program Description

Program allows user to enter new pub's information and update existing ones. The interaction between human and computer happens due to user interface coded in JavaFx. Data is stored in MySql server, images are stored on imagur.com.

2.2 Program Structure

2.2.1 See [1] section 2.2.1 and 2.2.3 for a list of classes with a description.

subsubsectionSee [1] section 5.1 for a list of methods with description.

2.3 Algorithms

loadFromDatabase(String townName) Loads pub list from database.

deletePub(Pub pub2delete, String town) Deletes pub from database.

processResults() Converts user input into pub object.

processEditResults() Converts user input in order to update pub object.

showFileChooser() Allows user to choose images to be added to the pub object.

invalidInputProperty() Checks user input if it is valid to create pub object.

showEditPubDialog() Shows Dialog pane to allow user to insert pub's information.

initialize() method from *EditAndDeleteController* Update user interface window to show proper information while user edit/add/delete pubs.

upload(File file) Uploads image to database.

2.4 The Main Data Area

Pub object has been used to store data for a single pub. PubList object has been used to store all the pubs available to the program at current time.

2.5 Files

Program uses two external libraries to set graphics and mysql-connector-java-8.0.15 to connect to database. These files may need to be updated or become deprecated, but they should not need to be actively changed or worked on to add features.

2.6 Interfaces

Having no access to the Internet does enable running the program but it makes impossible to connect to database and so program does not download or upload any kind of data.

2.7 Suggestions For Improvements

Making the user interface window resizable.

Adding feature to delete image from pub object.

After clicking "save" button, program shows enter screen and allows user to use it again.

2.8 Things Wo Watch When Making Changes

When adding a new component to any of .fxml files, check the layout created by that file (e.g. check if components do not stack one on the other).

Adding or deleting pub's characteristics will require to update .fxml files and methods which process the user's input.

2.9 Physical Limitations Of The Program

The compiler must run at least Java 1.8 version.

2.10 Rebuilding And Testing

All files have been saved in *SEgrp12DesktopApp* file.

To rebuild a system standard procedure for rebuilding the system should be followed.

When the program has completed running the tests the Run window will be opened and the information about passed and unpassed tests will be shown.

3 MOBILE

3.1 Program Description

A mobile application to provide a user with information about pubs in different towns and allows the user to create both planned and random pub crawls.

3.2 Program Structure

3.2.1 List of modules.

See [1] section 2.2.2 and 2.2.4 for a list of methods with a description.

3.2.2 List of methods

See [1] section 5.2 for a list of methods with a description.

3.3 Algorithms

linkToDatabase Links to database.

createLayoutDynamically Programmatically creates a layout based on variables, e.g. numbers of buttons.

sortListByTimes Sorts the list of pubs by time assigned to it.

onButtonShowPopupWindowClick Creates a popup window.

onKeyDown Goes back to the previous screen while changing variables appriopriately.

goToListOfPubs It sorts the filteredListOfPubs to contain the desired pubs and changes the screen to the listOfPubs screen.

onNavigationItemSelected Detects when a list item in navigation bar has been clicked and proceed to take appropriate actions.

onMapReady Displays the pub coordinates appropriately.

DownloadImageFromInternet Downloads the image from the Internet.

isNetworkAvailable Checks whether there is an access to the Internet.

onCreate Creates an activity and all of its components.

onCheckboxClicked Checks if the checkbox is ticked and proceeds to take an appriopriate action.

addPub Adds pub to the list of pubs in a Town object.

3.4 The Main Data Area

We have created ArrayLists to store a list of towns *ListOfTowns*, lists of pubs included in a crawl *PubCrawlList* and created a List of filtered pubs *PubFileteredList*. We have used objects such as: Pubs *currentPub* for the pub being looked up, *currentTown*- the town being looked up, *Pub*- for all the information regarding a pub and Town *Town*- all the information regarding a town.

3.5 Files

The application doesn't require access to any files.

3.6 Interfaces

Having no acces to the Interent does enable running the program, but it makes it impossible to connect to the database and so it doesn't get the pubs and towns information.

3.7 Suggestions For Improvements

Adding a pop up window when the program starts to inform the user whether there is no internet access.

Creating a limit to the size of the textView and buttons for pub/towns names and make it scrollable.

Improving the layout of the program for the *filteredPub* activity.

While starting a random pub crawl with the fileteredList being empty, display the error message asking the user to change filters instead of asking for inserting a greater number.

While starting an ordered pub crawl with the fileteredList being empty, display the error message asking the user to change filters instead of letting the user to go to the next addToPubCrawl activity.

Allow searching through a list of towns to take into account the number of equal letters from the whole word e.g. berystwyth search would display Aberystwyth before Borth.

Get the phone local time and use that to check off pubs in the ordered pub crawl.

Adding a possibility to assign a time to a random pub.

3.8 Things To Watch For When Making Changes

When adding a new filter option, the areas which have to be changed include the *linkToDatabase* function in the main class, *addPub* function in *Town* object, Pub constructor for *Pub* object and the *FilteredPub* class.

Changes made in any of the lists *pubCrawlList*, *pubFilteredList*, *pubList* can affect the other ones (e.g. the *pubCrawlList* is a copy of the *pubFilteredList* and the *pubFilteredList* is a copy of the *pubList*).

When adding a component to an activity, check if everything still fits in the screen.

3.9 Physical Limitations Of The Program

The user's mobile has to be an Android device. It is advised that it has an API up to level 26.

3.10 Rebuilding And Testing

Everything has been saved in a GroupProject file.

To rebuild a system a standard procedure for rebulding the system should be followed.

When the program has completed running the tests the Run window will be opened and the information about passed and unpassed tests will be shown.

REFERENCES

[1] *Group 12 Software Engineering Group Project* Design Sepcification. E Kalisiak, K T Todd, J Kus, SE.12.DS 3 Release.

DOCUMENT HISTORY

Version	CCF No.	Date	Changes made to Document	Changed by
1	N/A	07/05/19	Initial creation	EDK8, STT31, JAK55