Adam Hardie

STUDEWNT NUMBER

Mobile platform development   
 MHI322959-17-B  
Coursework report

Iain Lambie

Contents

[Project Overview 2](#_Toc509945702)

[Parsing the data 2](#_Toc509945703)

[Searching through the data 2](#_Toc509945704)

[Current Incidents 3](#_Toc509945705)

[Planned Roadworks 3](#_Toc509945706)

[Design: 3](#_Toc509945707)

[Elements used within this project: 4](#_Toc509945708)

[RadioButton 4](#_Toc509945709)

[ListView 5](#_Toc509945710)

[SearchView 5](#_Toc509945711)

[ViewFlipper 6](#_Toc509945712)

[SplashScreen 6](#_Toc509945713)

[TextView 6](#_Toc509945714)

[Google Maps API 6](#_Toc509945715)

[Design Principles 6](#_Toc509945716)

[Consistency 6](#_Toc509945717)

[Shortcuts 7](#_Toc509945718)

[Simplified error handling 7](#_Toc509945719)

[Testing Report 7](#_Toc509945720)

[Test Cases 9](#_Toc509945721)

[Summary 10](#_Toc509945722)

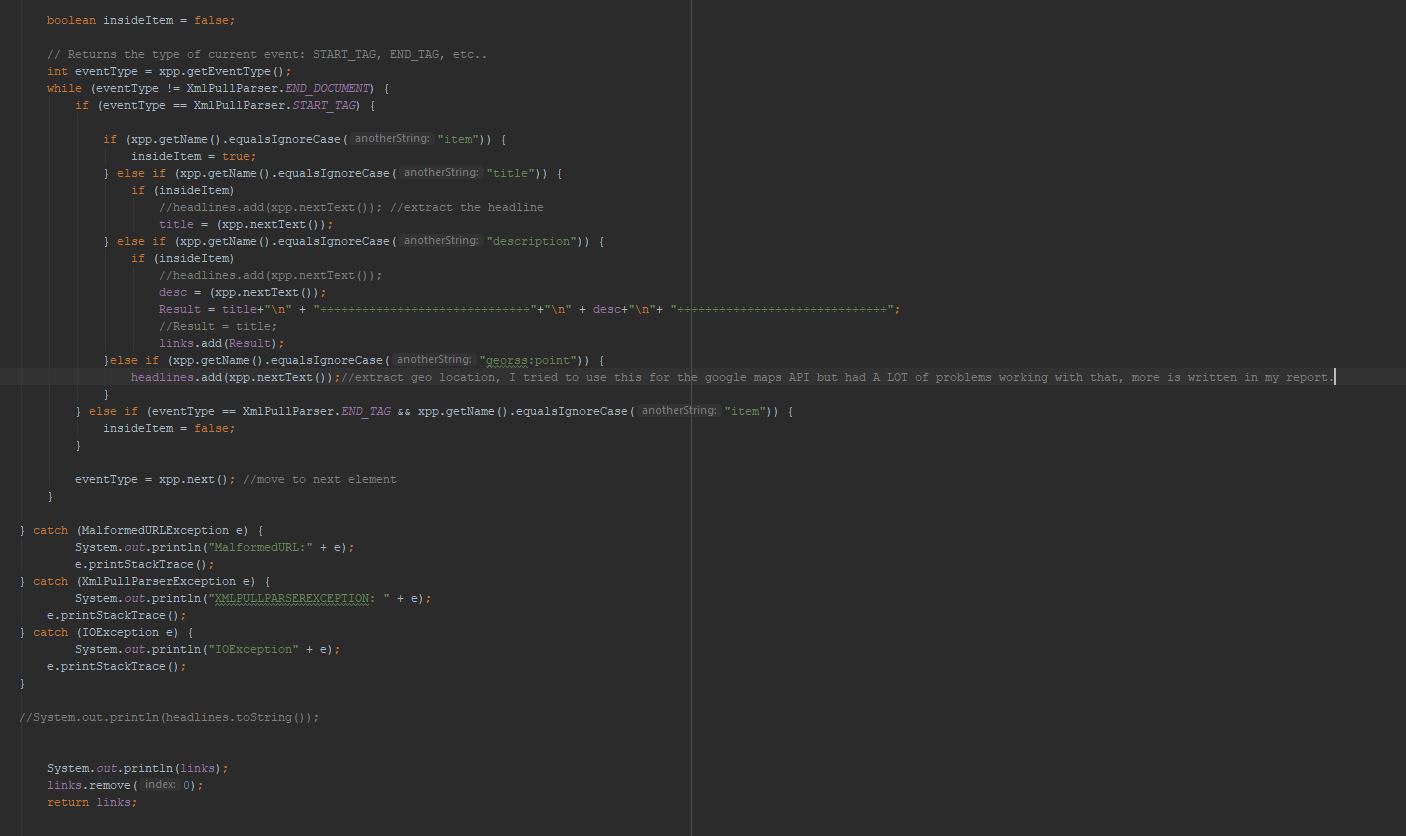
# Project Overview

The aim of this project is to develop an android application which will parse data from the Traffic Scotland RSS feed with the Current Incidents and Planned Roadworks data. One of the key requirement specifications it to allow users to search for a date or for a specific road which may or may nor have any current or planned roadworks.

The data must be parsed and stored in such a way that the application is able to access it and allows for the search functionality to be implemented.

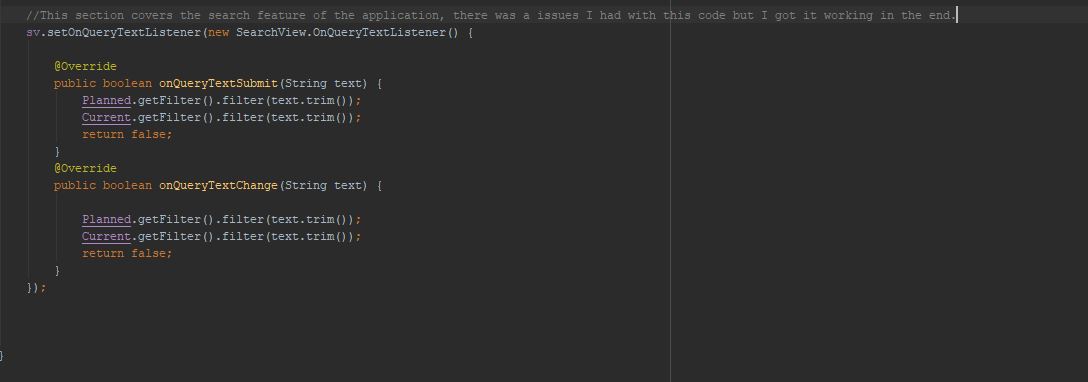
## Parsing the data

* There are two key classes which are responsible for parsing the data within this application RoadworksCurrentParse and RoadworksPlanParse.
* These classes use the XML Pull Parser which allows for XML data files or streams into java variables.
* Originally I attempted to use an array of objects to get this to work but after a lot of debugging, it was much more efficient for development to use String Array as I had much more success with this method over the previously mentioned.



## Searching through the data

* To search through the data within the ListView object, there was a couple of different methods used, in the end the SearchView was the best option.
* SearchView allows users to enter a value into a search field and query for their specified values.
* Another method which I attempted to use was just a simple EditText with a Button placed next to it, this method proved inefficient as there was a bunch of layout issues which I was running into with it.
* This is a snippet of what the SearchView code looks like.



## Current Incidents

• The Current Incidents feed can be found here:

• <http://trafficscotland.org/rss/feeds/currentincidents.aspx>

## Planned Roadworks

• The Planned Roadworks RSS feed can be found here:

• <http://trafficscotland.org/rss/feeds/plannedroadworks.aspx>

# Design:

One of the most prevalent aspects of mobile application design is the User Interface (UI), this one aspect can be the sole reason as to why users will use one application over any competitors who have applications that carry out the same, if not, similar tasks. During this project, designing both an easy-to-use and functional interface was at the top of the priority list as those are probably the biggest things about the UI which people look for in a mobile application.

To achieve the goal of creating not only a functional and aesthetically pleasing application interface but an easy-to-use to use one as well, there had to be research done into other popular mobile applications. During the research stage of the project there was a lot of knowledge which was gained by looking in-depth into the other popular applications which had a lot of user interaction. The results of this research phase produced blueprints of the prototype application which will be developed, these blueprints were an integral part of the development of this application as they helped keep the project on track.

By sticking to the following blueprints, it allowed for a relatively seamless development cycle. In android studio, the creation of user interfaces of the application is relatively easy with both the design view and the XML text editor. This project was created for the most part with the XML text editor as it was what I found was best to add certain constraints and variables to each element of the application interface. With the design view, it is easy to add in elements and certain restriction but the customization of on-screen elements was just not adequate when modifying small details about on-screen elements.

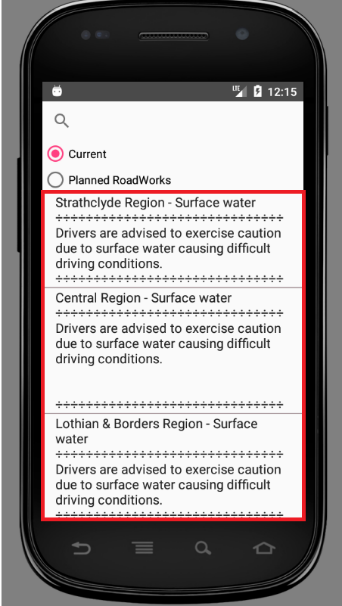
When researching other popular applications, there was a few categories which were focused on to try and remain relevant to the type of application that was being created in this project. That is, an application which will allow users to check up on the Traffic Scotland Current Incidents and Planned Roadworks RSS (Rich Site Summary) feeds. When researching other applications of a similar nature, the key aspects which were focused on were the elements which were used for each feature and how they worked in relation to other on-screen elements which created a much more user-friendly experience. The key features which I felt were best researched in this project were elements relating to the ‘Changing’ of which RSS feed the user is viewing, the ‘Search for a road’ feature and the display of the content from the RSS feed. The information gathered from this stage greatly assisted with the decisions made for the layout of this project.

## Elements used within this project:

In this project, there are many components which create the User Interface of the application, the biggest elements at play are the RadioButtons which control what specific RSS feed the user will have displayed, the ListView which will hold all of the items from the RSS feeds and will change depending on the user selection of RadioButton and a SearchView which allows users to search for a specific road within the Current or Planned roadwork RSS feed.

### RadioButton

* This component allows users to select one option from a given set of options, this element is especially useful if you want users to easily see all the available options on any specific interface, hence why I personally felt that it was a good option for this project as it will not only be clear to the user what options are available and which RSS feed is currently displayed. Because RadioButtons are mutually exclusive, they MUST be placed inside of a RadioGroup. The RadioGroup used in this project was to ensure that only one of the two options of RSS feeds available to the user could be selected, either Current Incidents or Planned Roadworks.



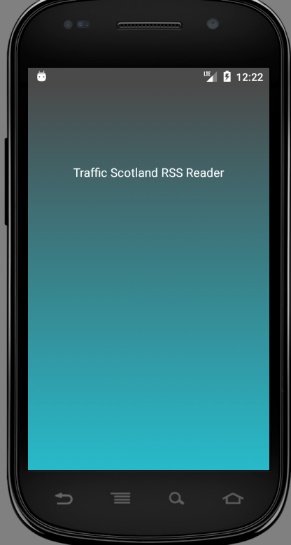
### ListView

* The ListView component is a view group which displays a list of scrollable items, the items which are included in the list are typically automatically assigned and inserted with an adapter which can pull content from many different sources, for example, the Traffic Scotland RSS feeds. The project as it stands right now works perfectly fine, but one enhancement which could have been made to the ListView was to build the list with the RecyclerView. A RecyclerView is much better for either larger data sets or data which frequently changes. Looking back on it, I believe this would have been better to implement instead of the ListView.

### SearchView

* A SearchView is a widget which gives the user the search functionality by entering a search query and submitting the request to a search provider. The SearchView was used in this project as it was the widget which I had the least amount of problems with. Originally, I was attempting to use just a plain EditText and a Button which would allow the users to search through the ListView and would return the specified results, but this method proved to very inefficient and wasn’t quite working how I intended so I switched to the SearchView. The SearchView in this application is positioned at the top of the interface to give the most screen availability to the user.

### ViewFlipper

* A ViewFlipper is essentially a way of animating between two or more different views which have been added do it, this is done with a switch statement which lets you setDisplayChild of the specific item within the view flipper, beginning at 0, increasing incrementally ‘flipping’ the view which the user has of the application.

### SplashScreen

* While the SplashScreen is not an individual component of the application, it is however a kind of ‘transition’ screen for the application to open. The SplashScreen of this application displays text informing the user what the application is, this is done with the TextView component which will be discussed below.

### TextView

* The TextView element in Android Studio is probably one of the simplest components available to the android device. It is a User Interface element which will simply display plain text to the user. It is used within this project to inform the user what the application they are opening is.

### Google Maps API

* Unfortunately, I was unable to get this functionality working in the application. My intention was to take the Users geo location in relation to the Current Incidents feed and draw a 25-mile radius around the user, which would take the geo locations of the Current Incidents and place markers around the pin of the user’s location, tapping on any specific pin would lay out a description of the incident to the user.

## Design Principles

The HCI design principles which were at play during the implementation of this project were the following:

### Consistency

* The ‘Consistency’ of a user interface simply put, means that consistent sequences of actions should results in similar situations in order to deliver a ‘consistent feel’ of the application and ease of use for the user. This was done well in this application, although, I will admit it could be better.

### Shortcuts

* As users become more and more familiar to any application, the begin to look for quicker ways of completing tasks, this is most common in computer applications with keyboard shortcuts, but mobile applications are no exception. Users will, much like desktop applications, look for quicker ways to complete tasks which, for example may take 5 or 6 clicks. To try and minimize the amount of user interaction required to complete certain tasks, the UI was made relatively simple to clearly define what each option on screen is and the outcome of each click on screen. However, one feature which I would have liked to implement would have been a search history in the SearchView element which would quickly allow users to search for their most travelled roads.

### Simplified error handling

* This design principle should be self-explanatory, the application should be designed in such a way that the user cannot make any fatal errors to the application. If by chance a user causes an error, the application should be able to handle this error and return a simple solution for the user to correct said error. Obviously, this goal of designing a fully bug free application is very ambitious and not likely, however, the design of this application was to create a simple, clear cut method of what the user should be doing and expected inputs at all points.

# Testing Report

One of the main forms of testing which I carried out on this project was Unit Testing.

Unit testing is the testing of individual units or a group of related units in order to see if the code which has been implemented is giving the expected results compared to the actual results.

During the Unit Testing Phase, I would work on individual segments of code, for example, the ViewFlipper. I would create a new project in Android Studio and create the code for a ‘working’ ViewFlipper, when I was happy with both the functionality and the ease of use with the view flipper, I transferred it over to the project application and implemented it into the application. Once I was happy with the implementation into the main project, I moved onto the next element of the application which is the SearchView.

Much like the ViewFlipper, I would create a new project and get all the functionality which I desired from the application, I would transfer it to my main project and implement it in a meaningful and easy to use way which would be effective at allowing users to search through the list view. The application which I created to test the SearchView was a simple SearchView element and a ListView which was populated with hard coded data, at this stage, the data was hard coded to reduce the scope of possible areas which could lead to any potential bugs or errors during this stage. Implementation of the SearchView into the main project was relatively easy as there were no glaring issues which popped up going from hard coded ListView data to parsed XML data which would populate the ListView during the OnCreate method in the MainActivity class.

When the SearchView, ListView and ViewFlipper were all working independently and were implemented into the main project, there wasn’t any issues which popped up and this came as a pleasant surprise and I am used to many errors popping up when modifying a project with code from another working project. Now that all the elements were in and seemingly had no errors, it was time to conduct Regression Testing.

Regression Testing is essentially the testing of a system after modifications have been made to the system, components or a group of related units to ensure that the modification to the main project is not causing any damage or imposing other modules to produce unexpected results. Regression testing falls under the category of Black Box Testing.

Black Box testing is a technique which will ignore the internal workings of a system and focuses mainly on the output of the system in question, in this case the overall application.

Another method of testing which was carried out on this project was Smoke Box testing. Smoke box testing is a system test which fits the purpose of identifying major flaws within a system. A good example of Smoke Box testing within this project would be with both the RoadworksCurrParse and RoadworksPlanParse java classes.

Within these classes there were Try/Catch blocks of code which were implemented to provide real time error handling, this was done to help prevent any major application bugs which would result in the termination of the application and allowed for any required changes to be made to the code in order to resolve the issues which would arise.

The final, and potentially most important part of testing during this testing phase was the Useability Testing. Usability testing is taking a look at the application design in the prospective of the client who will actually be using the application. This testing helps developers to figure out whether or not their application GUI is user-friendly or not. There are a few factors which went into this and the application not has a much better interface due to this helpful testing.

The biggest aspect of this testing was how easily was the client able to learn how to use the application and after learning how to use the application, how efficiently were they able to perform the tasks which they were set out to do. The two biggest changes which came from this testing was the removal of the ActionBar and the repositioning of the SearchView element. The feedback which was received was that there wasn’t enough room on screen for the users to properly read all of the list items which they wanted, the simple solution to this was to remove the ActionBar from the top of the application in order to create more on screen real-estate for the ListView items. The second piece of feedback which was received was that the SearchView was in an awkward position, originally the SearchView was below the RadioGroup which contained two RadioButtons, the simple solution to this was to reposition the SearchView to the top of the application, this created a much more visually appealing application. With this helpful data from usability testing, the application is in a much better standing and is much more user friendly.

The overall testing of this project was, I feel, carried out pretty well, this is in large part due to there not being any major issues which occurred during development of the application. If there had been more errors which were code related, I believe I would have had a harder time identifying what was the source of the errors as there isn’t much help for debugging built into the Android Studio IDE.

# Test Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Actual Result | Expected Result | Comments |
| Application Opens and user is presented the Main application |  | Application Runs successfully and the main application is presented | Working as expected |
| User clicks on Planned Roadworks radio button and the parsed data is output in the listview |  | The planned roadworks parsed data successfully populated the ListView | Working as expected |
| User clicks back on the Current Incidents radio button and the parsed data populates the listview |  | The current incidents data successfully populated the listview | Working as expected |
| User clicks the SearchView and is prompted with the keyboard for input of a specific road in the search hint |  | The seachview is clicked and the user prompted with keyboard and the hint for searching for a specific road | Working as expected |
| User Searches for the Road ‘m80’ in the Current incidents feed |  | No results found \*at the time of testing\* for any current incidents. | Working as expected |
| Users Searches for ‘M90 J9’ in the Current Incidents Feed |  | One result found \*at the time of testing\* M90 J9 Muirmont | Working as expected |
| User Searches the Planned Roadworks feed for ‘A90 Forth Road Bridge’ |  | User find the result A90 Forth Road Bridge for scheduled roadworks. | Working as expected |
| User searches for ‘Adam’ in the Planned roadworks |  | No result found. | Working as expected |

# Summary

This project taught me a lot about android development and the various aspects which need to be takes into consideration when developing an application for a touch screen device. There are a few things which I was unable to develop as I just didn’t have the time or prior knowledge to get fully functional. The following things are a few which I wanted to implement into the application:

Google Maps API –

I would have like to of had a Google maps api display inside of the application which took the users current location, drew a 25-mile radius circle around them and displayed all of the current incidents which were within that area. Allowing the user to click on a pin which would then display more information to the user about the Current Incident. I had a lot of trouble with attempting to implementing this functionality as I just wasn’t able to get it working no matter what U was trying. In future I would like to get something with this functionality working fully.

Calendar Search –

Something which I feel would have been a great implementation into this application would have been a calendar that allows users to search for roadworks by clicking on a specific date on the calendar and searching through the RSS data to find all roadworks which were planned for that date.

One thing I would change about the development process is to use a different IDE.