UNIVERSITY OF HERTFORDSHIRE

Faculty of Science, Technology and the Creative Arts

Modular BSc Honours in Information Technology

6COM0285 – Information Technology Project

Final Report

April 2012

4D

Richard King

Supervised by: Keith Dawkins

Abstract

The lack of new or updated HCI guidelines that can be widely recognised for use in mobile design have led to a mixed level of user friendly applications and other mobile interfaces. This report proposes a mobile user interface for Android smartphones which is based on sliding panels positioned around the outer edges of a home screen. These allow the user to access features such as favourite web pages and contacts, with a centralised news aggregator that enables the user to keep up to date with the latest news, sport and entertainment.

The report covers the entire development process of the interface design from the initial research to testing and evaluation. The research section focuses on how traditional heuristics such as Nielsen's Heuristics and Ben Shneiderman's '8 Golden Rules of Interface Design' have been built upon in attempts to create new sets of guidelines for mobile interfaces. In the design section each document or diagram related to the design stage found in the appendices is explained, detailing why I have chosen that particular method and why I have chosen certain designs. The implementation section is split up into the main elements of the design and describes how and why the features were implemented. Each section will contain a mini-evaluation which is then concluded in the 'discussion and evaluation' chapter.

Acknowledgments

A special thanks to:

- Steve Bennett for giving advice on several elements of the application
- Participants of the usability study
- Keith Dawkins (project supervisor) for his continued support throughout the project
- Lorna King for taking the time to read through my report
- Dr Martina Doolan (second marker)

Table of contents

1.0 Introduction

1.1 Background

In recent years there has been a huge surge in the popularity of smartphones. By the end of the year, Cisco predicts there will be more smartphones on the planet than humans (Trevor Mogg, 2012) But it's not just their popularity that's increasing, the wide variety of tasks that can now be carried out has also massively increased. You can now use a smartphone as a Sat Nav, to compare shopping prices, plan travel journeys, check the latest news, listen to your favourite music and much more. A survey revealed that people spend at least 90 minutes a day on their smartphone, with 25% of all smartphone uses preferring to do most of their online browsing on their mobile device (Aaron Smith, 2011).

One of the main problems with browsing the internet on Android phones is the number of disjointed jumps the user has to take to simply open, close and switch between web pages. To open a new web page, the user must navigate through 3 steps when the browser is initially launched, and to open any further pages requires at least 2 steps. To close a page the user must go through at least 2 steps. This is a contrast to browsing on a desktop which only requires one click to open a new page, and one click to switch between pages. It is also often not possible to see any information such as updates without jumping from the page you are currently on. 4D presents a new interface that allows users to quickly access web pages and other options effortlessly by simply sliding panels from the sides of the screen. It also features an RSS Reader which lists a number of news stories via an XML link. This is accessible via the main screen and its content can be changed via tabs on top of the RSS Reader.

1.2 Initial objectives

Main objectives:

- Create a user interface for Android OS incorporating touch screen control techniques
- Create an RSS Reader that sits centrally on the main page
- Create a customize page that allows the user to alter aspects of the interface such as colour and other features
- Test and deploy the application on an Android phone

Research objectives:

- undertake research into current HCI practices and specifically the uses for navigation interfaces for touch screen devices
- evaluate and compare traditional heuristics guidelines and how efforts have been made to adapt these for mobile interfaces
- what makes a good gestural interface
- patterns for touchscreens
- appropriate documentation for designing mobile interfaces

Software requirements and design objectives:

- create Use Case Diagrams
- write Requirements Specification Document
- construct Task Analysis
- use wire framing to create a paper based low fidelity prototype

Implementation objectives:

- create and code functionality and visual elements for RSS Reader, Customization page, panels and all other gestures

Testing objectives:

- write questionnaire and distribute
- conduct an heuristic evaluation
- evaluate and fix issues

1.3 Report structure

The first section Research, includes a Literature review and an evaluation into current mobile interfaces. The Literature review includes conclusions after each sub-section, with a full conclusion at the end of the review. This then leads on to the first main chapter, Chapter 1: Requirements and design, which discusses the requirements and design documents found in the Appendices. The second chapter, Chapter 2: Implementation discusses each main element of the application. The final main chapter, Chapter 3: Testing, discusses the techniques used and the results. Each section of the report contains an evaluation, with a summarised evaluation in the Evaluation and discussion section.

2.0 Research

2.1 Literature review

2.1.1 Introduction

Many of the traditional principles of interface design are used in mobile development today, but there has become an increasing need for these principles to be expanded to include guidelines for mobile design. This review will focus on two main themes, the fundamental principles of interface design, and the adaptions that have been made to update these methods. The review will be split into three main sections: a comparison between two of the most used principles in HCI: Nielsen's 10 Heuristics and Shneiderman's "Eight Golden Rules of Interface Design", an analysis of mobile interface design from two books, and an analysis from an article that has adapted traditional principles and created a set of guidelines for mobile design. Although the books and article are written in separate sections, there are many comparisons made between each section and cross-referencing is often used.

Heuristics

Background

Heuristics are a set of guidelines that can be used to test the usability of a user interface design. These are a set of recognised rules such as the popular Nielsen's 10 Heuristics or Ben Shneiderman's 8 Golden Rules of Interface Design. The main purpose of these are to identify problems with the design of the interface.

A heuristic evaluation can be conducted using use cases. It is also a good method to use at the start of the design stage. As the evaluation only requires one expert, rather than using potential end-users, costs are reduced and management is made easier as no venue is required and little planning is needed. A heuristic evaluation, therefore, can be completed in a matter of days.

Nielsen's Heuristics

Jakob Nielsen, P.h.D is a User Advocate and principal of the Nielsen Norman Group. He founded the "discount usability engineering" movement for fast and cheap improvements of user interfaces and invented heuristic evaluation. He is currently working on a new set of principles for mobile interfaces (Nielsen, 2011). For a full list of Nielsen's 10 Principles see APPENDIX A.

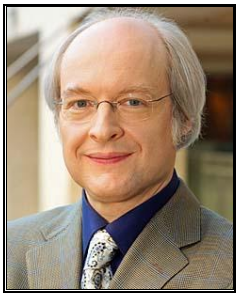


Fig. 2.1: Image of Jakob Nielsen

Ben Shneiderman's '8 Golden Rules of Interface Design'

Ben Shneiderman studied at University of Maryland and is founder of Human Computer Interaction Laboratory. His early work included database research and is also known for software engineering (Shneiderman, 2004). He came up with a set of rules named '8 Golden Rules of Interface Design'. These rules were based on his research and empirical data collected through various experiments. For a full list of the rules see APPENDIX B.

Fig.2.2: Image of Ben Shneiderman

2.1.2 Nielsen's 10 Heuristics and Ben Shneiderman's '8 Golden Rules of Interface Design' Comparison

Consistency and Standards

Both heuristics contain a similar principle to this. Shneiderman's rule states 'Strive for Consistency' while Nielsen's rule is 'Consistency and standards'. They both advise to clearly distinguish between elements within the design so users do not become confused as to the purpose of the action, and to keep these elements consistent throughout. Nielsen expands the principle by advising to maintain conventional standards of the specific platform.

Flexibility for expert users

Although worded differently, Nielsen's 'Flexibility and efficiency of use' and Shneiderman's 2nd principle 'Enable frequent users to use shortcuts' send out similar messages. Both advise to incorporate hidden shortcut keys, commands and facilities to meet the expert user's needs. Both principles point out that by doing this, the expert user's interaction speeds can be increased.

User feedback

Both heuristics contain a principle relating to user feedback. Nielsen's is 'Visibility of system status' and Shneiderman's is 'Offer informative feedback'. They suggest that every action should contain some level of feedback, with Shneiderman's principle detailing the level of feedback based on the frequency of the action.

Design dialog to yield closure

This principle specific to Shneiderman's rules does not have a corresponding match in Nielsen's Heuristics. The closest match would be Nielsen's 3rd principle 'User control and freedom', which states that there should be an 'emergency exit' to move from one state to another. Shneiderman's point explains that there should be a beginning, middle and end to a sequence and that the dialog given at the end of a sequence should give the user a feeling of completion and relief.

Error prevention and handling

Both principles offer very similar advice on this topic. Although not immediately obvious in their titles, Shneiderman's 'Offer simple error handling' and Nielsen's 'Error prevention' both emphasis the need to create a system that doesn't produce errors in the first place and also handles any errors effectively.

Minimise users memory load

Both heuristics feature similar principles on this. Nielsen's 'Recognition rather than recall' describes how objects should be made visible and that users should not have to remember information from one part of a dialogue to the next. Shneiderman's 'reduce short term memory load' advises that displays should be kept simple.

Reversing actions

Shneiderman's heuristics has its own principle for this, 'Permit easy reversal of actions', whilst Nielsen's Heuristics incorporates the suggestion of including undo and redo in its 3rd principle, 'user control and freedom'.

Support internal locus of control

This is a principle unique to Shneiderman's heuristics. It urges the system to be designed in such a way that users feel they are the initiators of actions rather than the responders. The closest match in Nielsen's Heuristics could be the 'user control and freedom' principle, yet its description states only about providing clear exit points from states and supporting undo and redo.

Help users recognize, diagnose, and recover from errors

As there are a total of 10 rules in Nielsen's Heuristics and only 8 in Shneiderman's, there are two extra principles to discuss. This first principle states that "error messages should be kept in plain English (with no code), precisely indicating the problem and constructively suggesting a solution". These last two principles of Nielsens Heuristics are highly focused on user friendliness and not an expert user.

Help and documentation

The last principle of Nielsen's Heuristics describes that although the system would be better if no documentation is needed to use it, it may be necessary to provide some level of documentation. The help should be focused on the users tasks and not be too long.

Conclusion

After comparing the principles of both Nielsen's 10 Heuristics and Ben Shneiderman's '8 Golden Rules of Interface Design' it has been made clear that there are many similarities between them. The differences between them have generally been due to wording or greater clarification on some points. There have, however been some other subtle differences, which I have summarized below:

- Nielsen's Heuristics are highly focused on user friendliness, with less guidance for meeting expert user's needs. This is evident in several principles and descriptions such as using 'error prevention' as a title, as appose to 'offer simple error handling' in Shneiderman's principles. It is also evident in the two extra principles both focusing on help for the user.
- Shneiderman's principles are more balanced, meeting both the needs of casual and expert users as evident in such principles as 'support internal locus of control' and 'enable frequent users to use shortcuts'
- Nielsen's Heuristics provide less clarity than Shneiderman's heuristics

In the following section I have reviewed two key books and one article on mobile interface design. I will be particularly focusing on how the authors have expanded on Nielsen's and Shneiderman's heuristics to incorporate guidelines for mobile design.

2.1.3 Review of Designing Gestural Interfaces (Saffer, 2008)

Introduction

This book by Dan Saffer, as the title suggests, focuses on the types of gestures that can be incorporated into, primarily, touch screen devices. It offers a large set of touch screen gesture patterns and discusses both the advantages and disadvantages of their implementation. It also offers advice on how to properly document your design, such as using use cases, storyboards and wire framing. In this review I will be discussing some of the key points the author makes on gestural interface design and documentation, and how these relate to the principles set out in Nielsen's Heuristic's and Shneiderman's '8 Golden Rules of Interface Design'.

Gestural interfaces

In his introduction to what makes a good gestural interface, Saffer (2008) explains that interactions should be made obvious and discoverable. He advises to provide feedback which should be made appropriate for the situation. The introduction ends with him describing the use of adaptive targets to guide the user in the correct direction, and providing the ability to undo actions. It is clear here that he has taken advice from heuristic guidelines such as Nielsen's Heuristics, for example in its first principle 'Visibility of system status' which includes "through appropriate feedback within reasonable time" (Nielsen, 1994) in its description. Providing the ability to undo actions is also set out in Nielsen's Heuristics and Shneidermans '8 Golden Rules of Interface Design'.

Attributes of gestures

In the section 'Attributes of gestures' a number of elements are presented. The duration of gestures is discussed such as how quick or slow they should be, and how long your finger should be held down before an action is triggered. Saffer (2008) questions how much pressure should be applied to the screen, and that the action may only require a slight tap to register.

The size of the screen is discussed, in particular how the screen size can be determined by its input method, for example if a stylus or finger is being used. Further on Saffer (2008) talks about the use of multi-touch technology, and how the use of more than one finger can be used to manipulate the interface, for example enlarging a box by stretching it.

He expands on his introduction regarding using undo and redo by advising to use these features on actions which are difficult to recover from. He ends this section by discussing states and modes, by providing clear paths through the system to indicate change of states. This statement seems to be inspired by Shneiderman's principle 'Design dialog to yield closure' that describes how actions should have a clear, start, middle and end point (Shneiderman, 2004) .

Appropriate gestures

In this section Saffer (2008) explains the appropriate use of gestures. He explains that gestures should match the behaviour of the system to gestures humans might already do to enable that behaviour. This seems to be inspired by Nielsen's principle 'Match between system and the real world, which explains the use of real world conventions and using concepts familiar to the user (Nielsen, 1994). He further discusses the need for caution in using complex gestures, noting that as we get older, our motor skills decrease. He also builds on his introduction by talking about screen size, for example advising never to place essential information below an interface element that can be touched, as it may be hidden by their hand.

Documentation

Saffer (2008) provides some useful techniques such as Use Case Diagrams, Task analysis and prototyping and less familiar methods such as wire framing, which would prove particularly useful in the design stage of my project. He advises to use a 'pixel-perfect' screen size when creating the wireframes to avoid cluttering the screen. Also discussed is the use of keyframes, which graphically illustrate major moments in an interactive gesture. They are used when something cannot be easily described, yet are not detailed enough for a storyboard. Another method mentioned is the swim lane framework, which combines a storyboard with states/views.

Conclusion

Overall Designing Gestural Interfaces by Dan Saffer has provided a comprehensive list of gestures for touch screen devices. He has explained in detail when to use these patterns and provided some helpful hints for their appropriate usage. Saffer (2008) has also given some detailed advice on how to document the design stage elements that incorporate gestures and screen size restrictions.

Dan Saffer appears to have used many principles outlined in Nielsen's Heuristics and Schneiderman's '8 Golden Rules of Interface Design' as a basis of his guidelines. He has provided advice on how to best utilise gestures to fit around these traditional heuristics and expanded on them to include advice on mobile interface design.

2.1.4 Review of Designing Mobile Interfaces (Hoober, 2011; Berkman, 2011)

Introduction

Unlike Designing Gestural Interfaces, this book by Steven Hoober and Eric Berkman focuses little on gestures but more on page layout, colours and other graphical elements in mobile design. It builds on traditional heuristic guidelines and expands these to incorporate mobile design. In this review I will be evaluating some of the advice that Hoober and Berkman have given on page layouts and how traditional heuristic principles have been used.

"Too much design is based on older paradigms for the desktop" Hoober (2011) explains in the introduction. This appears to be aimed at such principles as Nielsen's Heuristics (1994), which were not designed for mobile phone development. He goes on to say that "movements are now underway to design for the mobile experience first, before other forms of computing". Here he could be referring to Nielsen's announcement that he is working on a new set of principles for mobile design.

Hoober (2011) also argues that too many people are talking about working with smartphones, yet not enough attention is being focused on feature phones, despite the huge market share. On a recent blog on Jakob Nielsen's website in his early research with mobile design, a statement reads "Our first research found that feature phone usability is so miserable when accessing the Web that we recommend that most companies don't bother supporting feature phones"(Nielsen, 2011). This may mean that whilst Hoober may argue that not enough attention is being given to feature phones, that unless web access improves on these devices, developers may need to focus more attention on smartphones.

Page layout principles

In this section there are many pointers to traditional heuristics. Hoober (2011) emphasises the importance of consistency, with the placement of elements on a page not just consistent with the application itself, but the whole operating system. This appears to stem from Nielsen's principle 'Consistency and standards' which states in its description to "follow platform conventions" (Nielsen, 1994). Hoober (2011) then goes on to advise that elements such as titles, buttons and notifications should be at the same position on every page. He also expands to talk about using valuable screen real estate wisely, where every pixel is important.

"Use consistent and simple navigation elements. People have limits to the amount of information they can store in their short-term memory. Therefore, they automatically filter information that is important and stands out" repeats both Nielsen's and Shneiderman's principles. Hoober (2011) even refers to Nielsen by stating "consider how users will view your page when plotting content. Generally, users will look for high-priority information in the upper left of the content area (Nielsen 2010).

Titles

The use of consistency is again repeated here with the advisory to use titles in a consistent manner. It is again pointed out that the colours and font should be similar to that of the operating systems design, despite the fact that you may be using significantly different styles to brand your product.

Error messages

It is recommended to "avoid excessively harsh error messages, and other things which may confuse or annoy typical users" (Hoober, 2011) which is also expressed in Nielsen's Heuristics principle 9 'Help users recognize, diagnose, and recover from errors' where it describes that error messages should be kept in plain English and not computer code (Nielsen, 1994).

Back key

It is advised that the dedicated hardware 'back' key is used to move backwards in an application as users will be accustomed to it. This reflects on Nielsen's principle 'Consistency and standards' regarding using platform conventions.

Hoober (2011) summarises the page layout section by emphasising the need to pay attention to small mobile screens which have little screen real estate and therefore every pixel counts.

Components for Control and Confirmation

This sections most significant point states "When costly human error is possible, we should create modal constraints and decision points to prevent errors before they happen". This fits in with Nielsen's 3rd principle 'Error prevention', which explains that with careful design, the system should be implemented well enough to cause little error. It reiterates this message further on when stating 'Some mistakes are caused by us, others by objects in our environment. But many mistakes can be prevented!' (Hoober, 2011).

As this section develops, Hoober (2011) expands on traditional heuristics even further and offers some advice solely related to mobile design. He explains the use of control, respecting the user input while protecting against user error, data loss and unnecessary decision points. It is described as a key part of mobile design.

Feedback

The section ends on how to use feedback in mobile design. Some of the advice given includes designing actions which have immediate feedback, creating changes of states, and using confirmations when the user is at risk of important data being lost.

Conclusion

Overall, Designing Mobile Interfaces by Steven Hoober and Eric Berkman attempts to expand on traditional heuristic methods by incorporating mobile design elements, primarily focusing on aesthetics and navigation. This involves taking extra measures such as resolving screen size limitations and taking environmental factors into consideration. Hoober and Berkman recognise the need for a new set of principles specifically for mobile design that can be widely recognised, and have made a promising start already.

2.1.5 Review of Guidelines for Handheld Mobile Device Interface Design (Gong, n.d; Tarasewich, n.d)

Like Designing Gestural Interfaces (Saffer, 2011) and Designing Mobile Interfaces (Hoober, 2011; Berkman, 2011) Jun Gong and Peter Tarasewich have taken traditional heuristic guidelines and built on them to provide advice on how to design for a mobile interface. However, they have condensed this advice into a more structured set of principles, and although they borrow heavily from Sneiderman's heuristics, due to the similarities between them, these modifications may be all that is needed to create a full set of mobile design principles. For the full review of Guidelines for Handheld Mobile Device Interface Design please see APPENDIX H.

Conclusion

Literature review conclusion

After reviewing both traditional heuristic principles and the attempts by some to update these to incorporate mobile interface design, I have developed an understanding of many fundamental issues to be aware of when designing my own interface. It has enabled me to find a balance between using such trusted principles as Nielsen's Heuristics as a basis whilst being able to enhance the user experience further by considering mobile specific advice.

2.2 Evaluation of current mobile interfaces

Iphone vs Android interface

I conducted an evaluation of both the top rival smartphones most up to date operating system's interface elements, focusing on the main features that both platforms share and comparing them by looking at the advantages and disadvantages of each. I also created a table that compares the different types of gestures used to interact with the interface. By conducting this evaluation it allowed me get a good idea of what designs were working best and gave me inspiration for my own ideas.

Lock screen

Both lock screens require the user to swipe the screen sideways to unlock the screen. The main difference being on Android's lock screen you can swipe anywhere to the right to unlock the screen, but on Iphone you have to swipe at the designated area. Although Android's way enables the user to possibly perform the action faster, Iphones method is more clearly defined as to what you need to do, and therefore is more user friendly. Also, on Iphones unlock screen you can access notifications directly from this screen without first unlocking the screen. This enabled the user to perform this action quicker, but privacy is a possible issue here.

Notification drop down

These feature on both phones, and act in an almost identical way. The content on Iphones drop down appears to be more dynamic, utilising the swipe gesture to horizontally scroll through widgets such as weather, and allows for automated scrolling of information such as the markets. Androids notification menu offers less in the way of utilising gestures, but does have a set of on/off symbols which allow the user to quickly turn features off and on such as Wi-Fi, GPS and vibration. The notifications bar can always be accessed and both the swipe and drag gestures work very fluidly.

Buttons and icons

The main difference between the button design on both platforms is that Iphones buttons are all the same square shape. Androids icons taken on different shapes depending what icon is for. The hit zone (the area in which if the user touches will trigger an action) is more defined on the Iphones buttons, and therefore may be more user friendly because of this. Androids hit zones are less defined and may lead to users having to press some buttons twice to trigger an action. The size of icons is much more customizable on Android, letting the user resize buttons and widgets to their liking and arranging these where they desire.

Gestures

Many of the gestures required to navigate around the interface are very similar on both platforms. Below is a table that shows various actions the user can take, their required gestures for both Iphone and Android and how effective each one is.

Action	Gesture for Iphone	Gesture of Android	Effectiveness on Iphone (1-5)	Effectiveness on Android (1-5)
Access main menu pages	Swipe to the left	Swipe to the left	5	5
Access drop down notification bar	Swipe downwards or drag downwards	Swipe downwards or drag downwards	5	5
Access applications	Tap to select	Tap to select	5	4
Scroll through contacts and other similar menu options	Swipe, hold and drag and tap to stop	Swipe, hold and drag and tap to stop	5	4
Zoom in	Pinch to zoom	Pinch to zoom	4	5
Highlighting text	Tap, hold and drag	Tap, hold and tap select from options	5	4
Switch between recently used applications	Double tap home button	Tap and hold the home button	4	4

3.0 Chapter 1: Requirements and Design

3.1 Use Case Diagram

I created a Use Case Diagram to demonstrate various actors and what type of tasks they will carry out in the application. I also included the touch screen gestures that would be used to allow the user to navigate the interface. Each actor had a different agenda so their journey through the navigation space would each be unique, although they would use universal gestures to ensure consistency throughout.

To produce the Use Case Diagram I used software called ArgoUML, an open source UML modelling tool which I have used successfully in past modules to create UML diagrams. I found this tool easy to use as it includes most UML diagrams and offers drag and drop techniques to flexibly place the components to where you want them to be.

3.2 Initial Use Case Diagram

With reference to APPENDIX C **,** you can see that there are a total of 5 actors, not all of them being people. The two users, User 1 and User 2 both have different agendas and therefore complete a different set of tasks. User 1's goal is to check for updates using the RSS Reader and check their email. This use case demonstrates the task of switching between panels and using the shake feature to update the RSS Reader. User 2's goal is to change the colour layout of the application. This Use Case demonstrates switching between panels, the shake feature and accessing the customize page. Both Use Case tasks interact with other outside actors, namely the XML feeds and web pages. The XML feeds are in the form of general headline news, entertainment news and sport news.

3.3 Updated Use Case Diagram

A second Use Case Diagram was created to reflect the changes made after usability testing (See 'Changes' section in the 'Implementation' chapter as well as Chapter 3: Testing).

With reference to APPENDIX D, you can see that there are a total of 5 actors, not all of them being people. The two users, User 1 and User 2 both have different agendas and therefore complete a different set of tasks. User 1's goal is to slide the first panel to access Google's homepage, then shake the mobile device to return the panel to its original position. This use case demonstrates the task of switching between panels and using the shake feature to close open panels. User 2's goal is to view a story from the RSS Reader and change the news feed by flicking between the feed tabs. This Use Case demonstrates using the customize panel and accessing viewing a news story.

3.4 Task analysis

I next created a Task Analysis comprising of a set of activities the user will undertake to complete a desired objective such as customising the user interface. In parts of the task flow, decision boxes were used which lead to different path ways through the navigation space. This allowed me to map out the design and incorporate measures such as providing back buttons to return to pages and main menus, without traversing a number of unnecessary pages.

Like the Use Case Diagram, the Task Analysis diagram was constructed using ArgoUML. With reference to APPENDIX E**,** the task carried out by the user is to customize the main screen of 4D. The rectangular boxes represent when the user uses a touch screen gesture to interact with the interface. The oval symbols show the outcome of the user's interactions. The diamond shape represents that a decision needs to be made, the Boolean values of Yes and No. These evoke different responses from the system and lead the user on a different path depending on their selection. When designing this Task Analysis, I needed to envisage where the user would like to be taken and using the quickest route. I included a confirmation dialog box so the user can change their mind before making any changes.

An updated Task Analysis for a user changing the background colour was made to reflect changes made. This can be found at APPENDIX F.

3.5 Wire framing

The final part of my design stage was to produce paper based wire frames. This helped me view the main features the application would include and focused on mainly functionality and content. I used a 'pixel perfect' screen size representation to ensure I did not clutter the screen and could arrange the interface having in mind that this would be the canvas size I would be working with in my implementation stage. The main features the wireframes would show are the controls, conditional objects and states, and gestures required to navigate the interface. I decided to produce wireframes after reading advice from Saffer, D.S., 2009, *Introducing Interactive Gesture,* Sebastopal, CA, which as previously mentioned, also gave advice on other documentation suitable for gesture based design.

With reference to APPENDIX G this initial wire frame shows the gestural action at the top of the page, and the subsequent action as a sketch underneath. The available actions on each page are detailed under each sketch and some annotations point to significant elements of the interface. The first page shows the sketches demonstrating the panels that lead to applications. The main screens interface elements such as the buttons and RSS Reader are shown, and a sample of some proposed applications that the user may wish to navigate to.

APPENDIX H shows the new wire frames that were created to reflect changes made after usability testing had been conducted.

The sketches show what the interface will look like when the user uses the customisation page to change the background colour of the main screen. The customisation page allows the user to select from a selection of options: Screensaver, Colours and News. Each button represents a hit zone where the user can tap to navigate to a separate page where the options can be set. In this case the user has selected to change the colours, and the user is presented with a colour pallet in which a number of different colours are available. They can use either just the 'tap to select' gesture or a combination of the 'tap to select' gesture and the 'slide' gesture to pick a colour from the pallet.

The next wire frame shows the interface elements when a user chooses to change the RSS Reader from its current selection to another new feed. Finally, the last fire frame demonstrates the user opening the contacts panel and selecting a contact to phone.

3.6 Requirements Specification Document

To detail what 4D will do in a form which is suitable for non-computer experts I created a Requirements Specification Document. This document will help me when designing the software because it outlines the functional requirements and non-functional requirements for each of the core elements that will be included in 4D. It also gives an outline of the purpose, scope and overview of the project. I decided to create an RSD because I felt it was the best way to deliver the system requirements in a form which can be understood by non-experts and is recognised as an industry standard documentation.

The Applications Perspective section briefly describes the similarities and influences the interface has with other software currently on the market. The similar gestures used in the current OS will help the user become more quickly familiar with using the interface (see Usability study on 4D using Nielsen's Heuristics section for more details).

The Functional Requirements section details what features are required in order for the software to function. A brief description of the main elements of the interface is given, along with a rationale explaining why this feature is necessary. The requirements that would both be necessary and beneficial for these elements to work properly are also outlined. Software and hardware requirements are also given in a separate section.

Lastly, non-functional requirements are listed. These range from speed of connections to the screen size resolution, and are not required for all parts of the software to work. However, meeting these type of requirements would generally mean the user will have an enhanced experience. In order to derive these non-functional requirements I had to think outside the box, looking at how external applications and network connections would affect the efficiency and performance of 4D. Please find the Requirements Specification Document in APPENDIX I.

3.7 Chapter 1 Evaluation

In the design stage I have used recommended design documentation methods outlined in Introducing Interactive Gestures (Dan Saffer) such as use case diagrams, task analysis and wire framing. Whilst I have used use cases and task analysis in previous modules, wire framing was a new concept to me and I found it particularly useful in designing a 'pixel perfect' layout. The disadvantage I found when using wire framing is that, although it allowed me get an idea of the positions and sizes of elements that would be placed on a page, it doesn't show detailed design elements such as colour overlays and gradients.

I found that using the software argoUML to create the use case diagrams and task analysis was easy to use and I was able to create clear and meaningful diagrams. Using use cases and task analysis gave me an understanding of the tasks and navigation flows the end users would go through in the application. It enabled me to identify where to incorporate features outlined in heuristic guidelines such as providing adequate feedback.

4.0 Chapter 2: Implementation

4.1 Panels

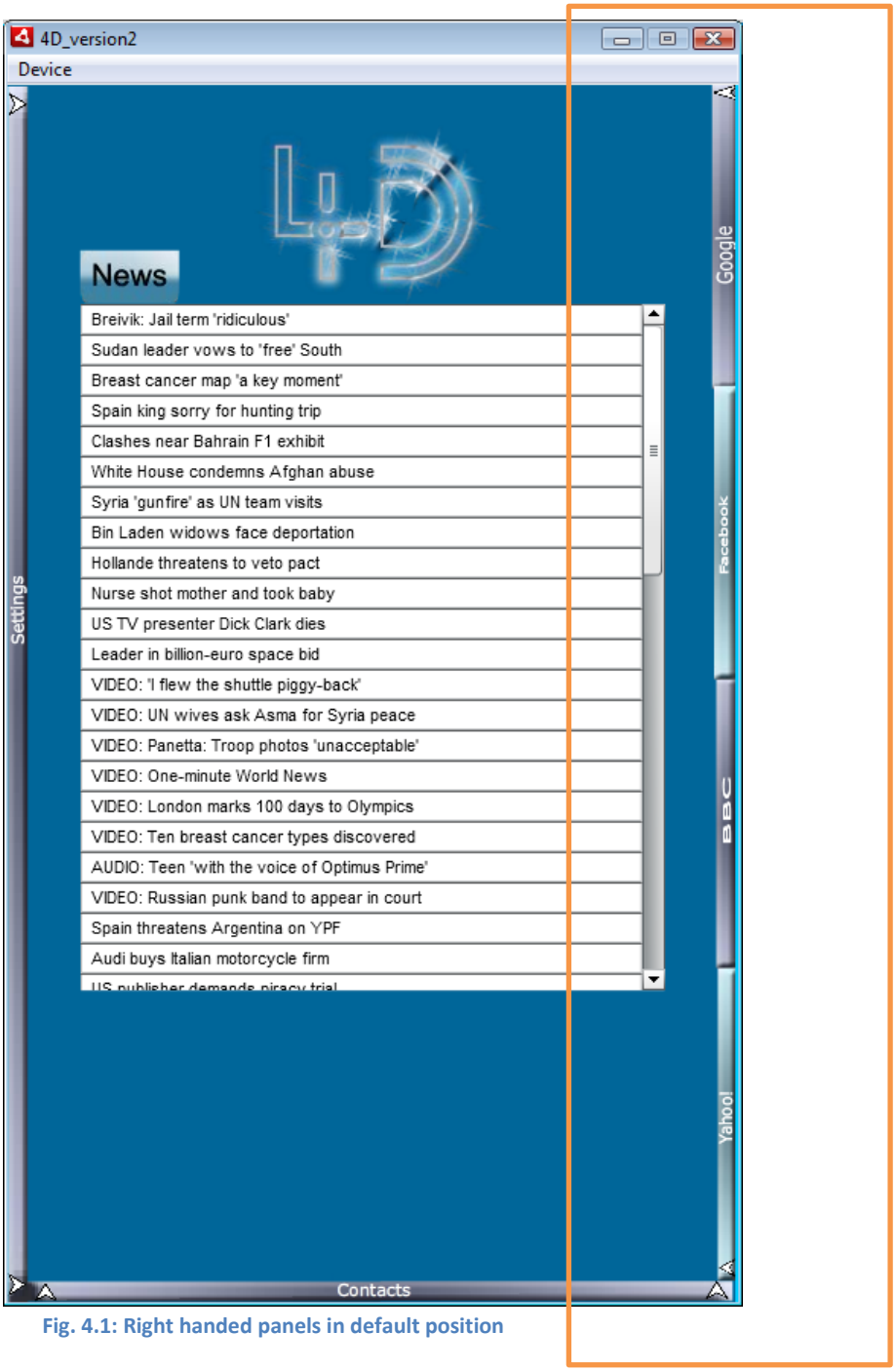


Fig. 4.1: Right handed panels in default position

I divided all of the elements I needed to implement into sections so it was easier to manage. This allowed me to focus on one aspect at a time and manage my time more effectively. The first part of the interface I started work on was the right handed panels.

After testing that the slider dimensions were suitable for touch screen interaction on the mobile phone I created the first panel outline. At this point I decided to focus on implementing the functionality of the panel, returning to the aesthetics at a later point. I made this decision in case changes arose with the programming which could have led to the physical shape of the object needing to be changed.

As outlined in the design section, the final decision as to what method would be used to utilise the gestural interaction of the panels was to incorporate a drag feature. However the problem faced was that the user should not be able to simply drag the panel anywhere, but only on a horizontal path with designated stop points.

In order to accomplish this I needed to find a way the panel could be confined within a specific area. One solution I had was to code in the exact dimensions of the area the panel could move in, but this would prove time consuming. Another solution was to use a motion tween guide (a feature in Flash that allows an object to move along a set path) but this was more suitable for animating. After investigating the best possible solution I came across a method used conventionally to drop and drag objects within the confines of a shape, such as a rectangle. I found this to be commonly used in creating jigsaw puzzle games and other drag and drop games. I decided that I could use this method to solve my solution. Fig. 4.2. illustrates this concept.

Fig.4.2: Inner rectangles are confined by a larger rectangle

To adapt the method to my design, I needed to make the outer rectangle transparent and shape it such that there would be no vertical movement. This would effectively mean the panel would only be able to move within the confines of the rectangle, and I could easily shape it to fit the required dimensions.

```
// Enable drag
imageHolder1.addEventListener(MouseEvent.MOUSE_DOWN, DragImage1);
function DragImage1(event:MouseEvent) {
        //  apply the boundsRect when they drag
        imageHolder1.startDrag(false, boundsRect);
```

The above code snippet allows the user to drag the panel. ImageHolder1 is the instance name given to the panel. A mouse down event listener is created with the DragImage1 function. I have used the conventional Mouse Down event used in for desktop software programming instead of the dedicated gesture event as they slow down the software due to high memory consumption. This is especially important when designing for mobile devices, as memory and processing power is often limited.

```
// Stop drag
imageHolder1.addEventListener(MouseEvent.MOUSE_UP, DropImage1);
function DropImage1(event:MouseEvent) {
        imageHolder1.stopDrag();
```

In the stop drag section another event listener is created, this time the mouse event Mouse_Up, which listens for when the user has taken their finger off the object.

To ensure that the panels functioned properly, the positioning of both rectangles had to be extremely precise. A difference in one pixel could cause conflicts with other objects. I noted this as a disadvantage of using this particular method.

After testing the first panels functionality on the mobile phone I next designed the sliders visual interface in Adobe Photoshop CS5. I decided to use Photoshop to create the elements that require aesthetic appeal due to the increased level of options.

To create the slider I used a transparent background and applied several effects to the shape. The first of these was a gradient overlay which gave the slider a natural feel and look. I made the darkest area of the gradient in the middle of the shape, to give the user a subtle hint to slide it from the middle area, to avoid accidently hitting an adjacent slider. The next effect I added was a Bevel and Emboss. This gave the slider a feeling of depth, almost making the shape appear 3D. The penultimate effect I applied was drop shadow and lastly a colour overlay. As my entire interface is going to centre around the colour blue, my design for the right hand panels was to use an alternating mix of dark and light shades of blue. The reason for choosing an alternating pattern is to ensure the user acknowledges where each slider ends and where the next one starts.

Fig. 4.3: Enlarged image of panel slider showing gradient, bevel and emboss and colour overlay

Now that the panels drag functionality and visual elements were complete, it was time to add web pages into the panels. This proved to be one of the most challenging parts of the project. I discovered that the ability to integrate web pages into Air for Android applications was a new feature, and that Action script were still working on maximising its capability. I therefore had to work with code that had limited expandability, yet produce the outcome I was looking for. I used online forums to gather knowledge of how programmers had expanded the capability of the code using 'work arounds'.

```
webView.stage = this.stage;

webView.viewPort = panel1.placeholder.getBounds(stage);

webView.loadURL("http://www.google.co.uk/");
```

The code above shows how Google's homepage was integrated into the first panel. An invisible placeholder object is positioned on the panel, where its outer bounds edges are calculated every time the panel is moved.

Below shows Google's homepage integrated into the first panel

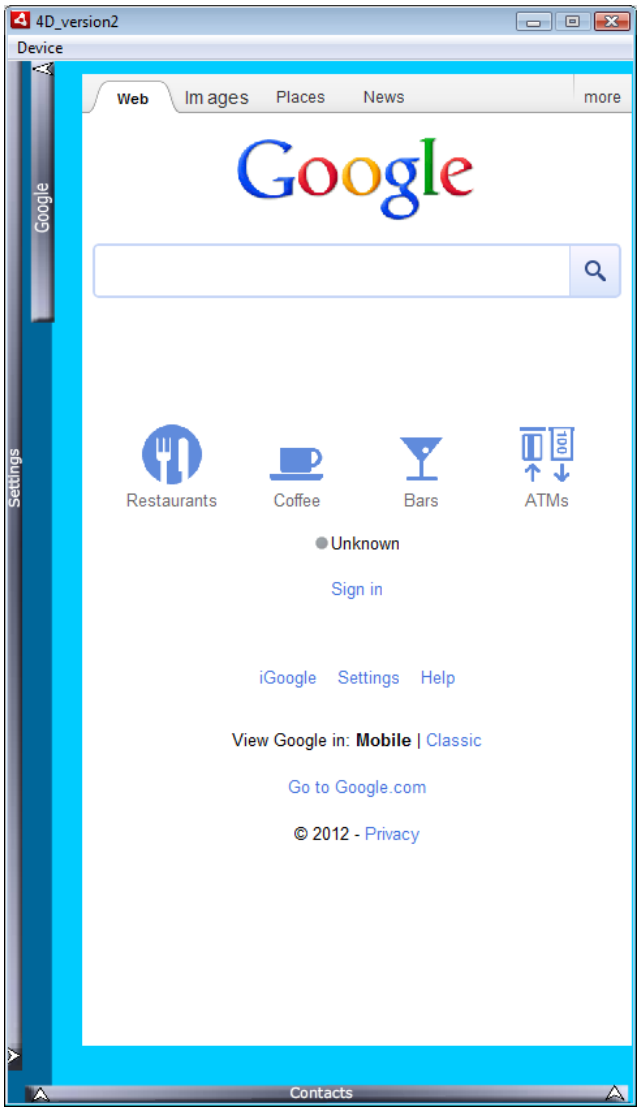


Fig. 4.4: Google's homepage integrated into panel 1

Below shows Facebook integrated into the second panel.

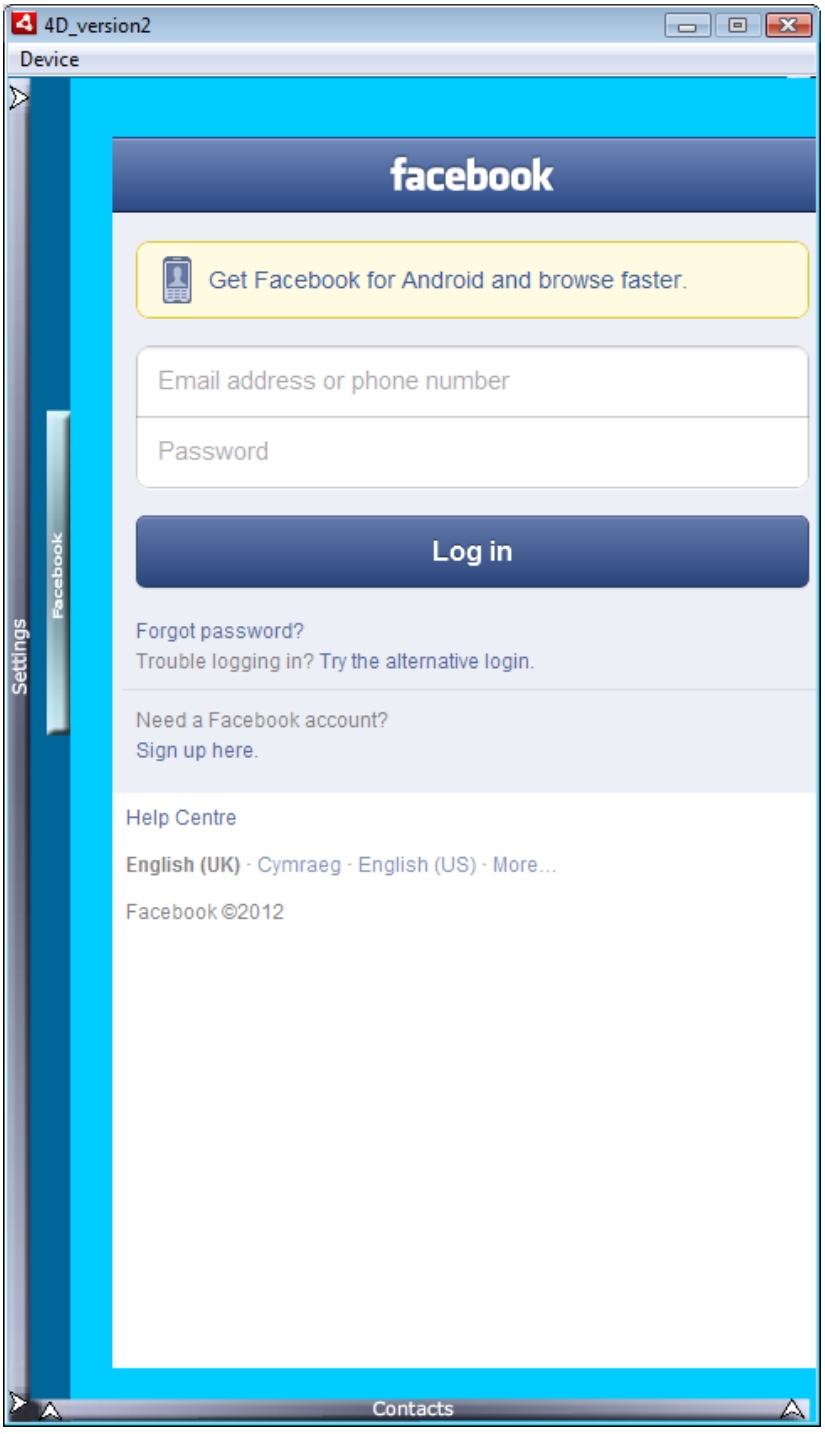


Fig.4.5: Facebook's homepage integrated into panel 2

The image below shows the BBC homepage integrated into the third panel.

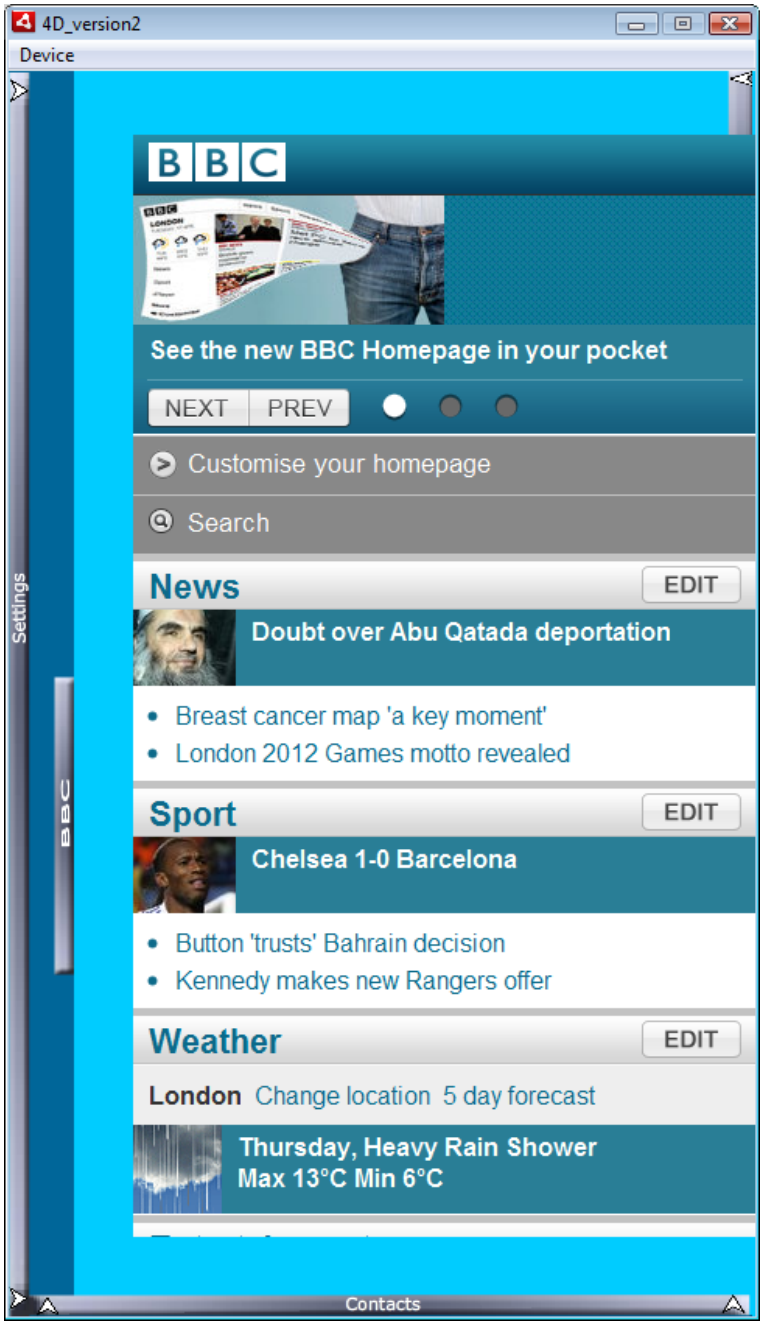


Fig.4.6: BBC's homepage integrated into panel 3

The image below shows the final page Yahoo, integrated into the fourth panel.

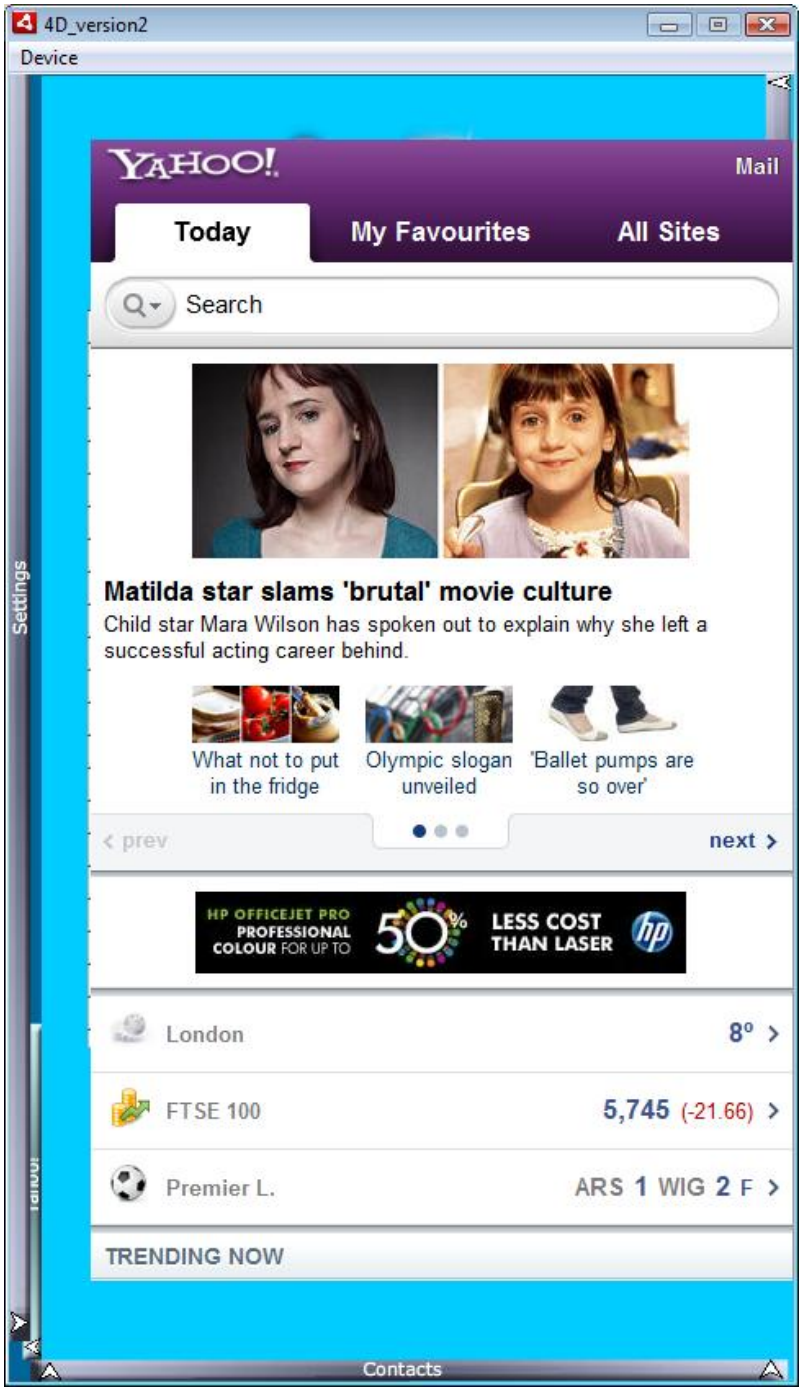


Fig.4.7: Yahoo's homepage integrated into panel 4

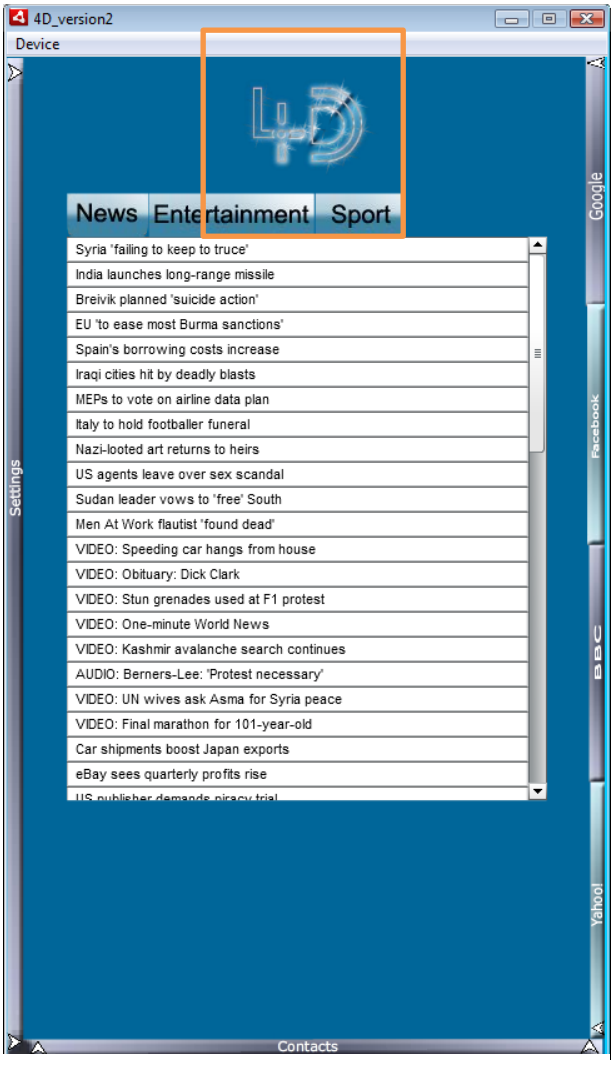
Now that the right handed panels were completed, it was time to move on to the first of two main elements of the main page, the 4D spinning logo.

4.2. 4D Spinning Logo

The 4D logo will feature in the central upper area of the main screen. It will not only serve as an aesthetic element but also as a screensaver, which can be turned off in the settings menu to save battery consumption. The logo was created using font creator Flaming Text (2012) which I have used successfully in other modules to create visually pleasing fonts. I chose a font which had a modern feel to it, which was catching to the eye and worked well as a logo. I added a blue tint and created a transparent background. The font size was also important; if it was too small the user would not be able to see its animation effect. However if it was too large it would take up too much room on the page, and with the limited screen real estate as described by Hoober (2011), I had to find a balance.

The original function of the spinning logo animation was that it would play every time the user shakes the mobile phone, to update the RSS Reader. It was then changed following usability testing to the screen saver. I used a 3D rotation tool in Flash which enabled the logo to rotate 360 degrees clockwise.

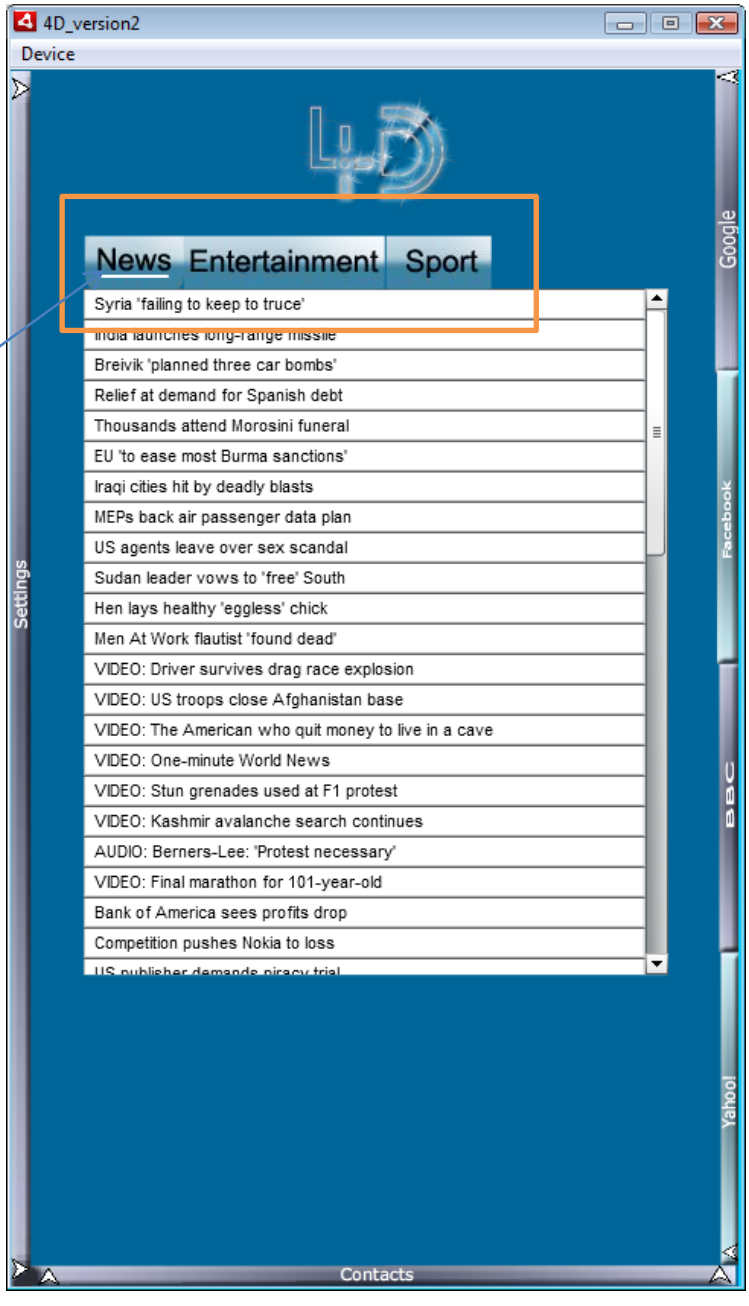
Figure 1 Fig.4.8: Spinning logo



Now that the logo was in position, it was time to move onto the second element of the main page, the RSS Reader.

4.3 RSS Reader

One of the main elements of 4D is the integrated news aggregator that sits centrally on the main screen. The reader allows users to check the latest news by scrolling through a list of stories that have been obtained through an XML file. They will then be able to read about the story further by selecting one which displays an extended version of it at the bottom of the page. Users will be able to switch between news, entertainment and sport feeds by using the tabs located above the list. The tabs visual design is similar to buttons used in the menu to ensure I maintained consistency throughout the application.



White line indicates which news feed the user is currently on. This meets Nielsen's 'Visibility of system status' principle.

Figure 2 Fig.4.8: RSS Reader tabs

```
var rssURL: URLRequest = new
URLRequest("http://newsrss.bbc.co.uk/rss/newsonline_world_edition/front_page/rss.
xml");
```

The above code sets up a new URL in the form of an xml file. I chose a feed from the BBC News as a default setting.

```
function rssLoaded(evt:Event):void {
        rssXML = XML(rssLoader.data);


        for (var item:String in rssXML.channel.item ) {
                liLog.addItem({label:rssXML.channel.item[item].title});
```

In the above function a loop will go through all the story list of items, and print the title into the list box.

The second function allows the selected story to display in full in another text box at the bottom of the page. The selectedIndex is the numerical number of the specific story that's been selected. The description is the story that is associated with the title in the liLog, and is what will display in the taLog (the text box at the bottom of the page).

```
function selectLog(evt:Event):void
{
        taLog.text = rssXML.channel.item[evt.target.selectedIndex].description;
}
liLog.addEventListener(Event.CHANGE, selectLog);
```

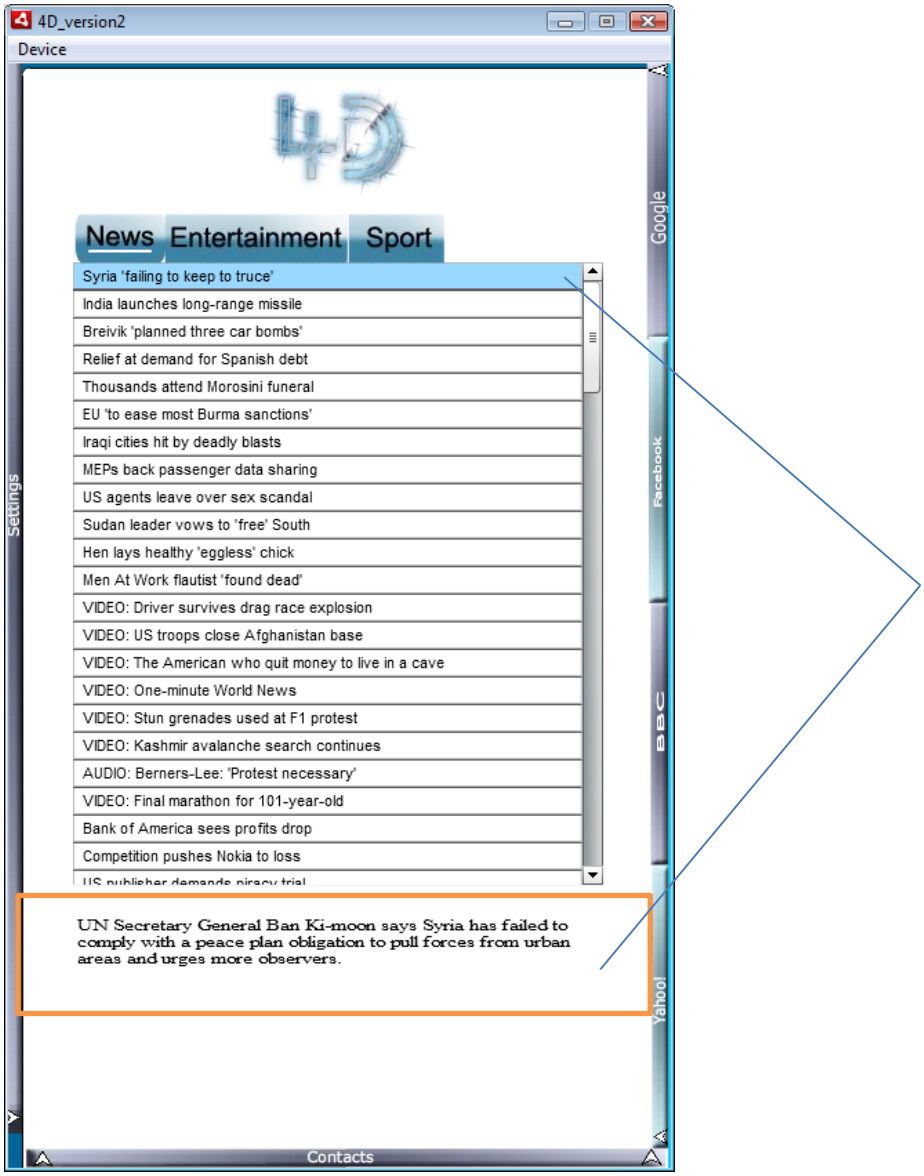


Figure 3 Fig.4.9: Lower box showing extended detail of the currently selected story

The RSS Reader finalises the last element to be implemented on the main page. The next task was the implementation of the settings panel, which is situated on the left hand side of the main page. The first setting I worked on was the colour picker.

4.4 Colour picker

The first option in the customisation panel is the ability for the user to change the background colour of the main screen from a colour palette. The first stage was to create a button that led the user to another page, where they are able to select the colours from. I applied a gloss to the button and chose a warm blue colour overlay which fitted well with the rest of my designs. I also rounded the edges of the button to create a smoother feel. In order to distinguish between when the button is being pressed and when it is not I created two states, an 'up' state and a 'down' state.

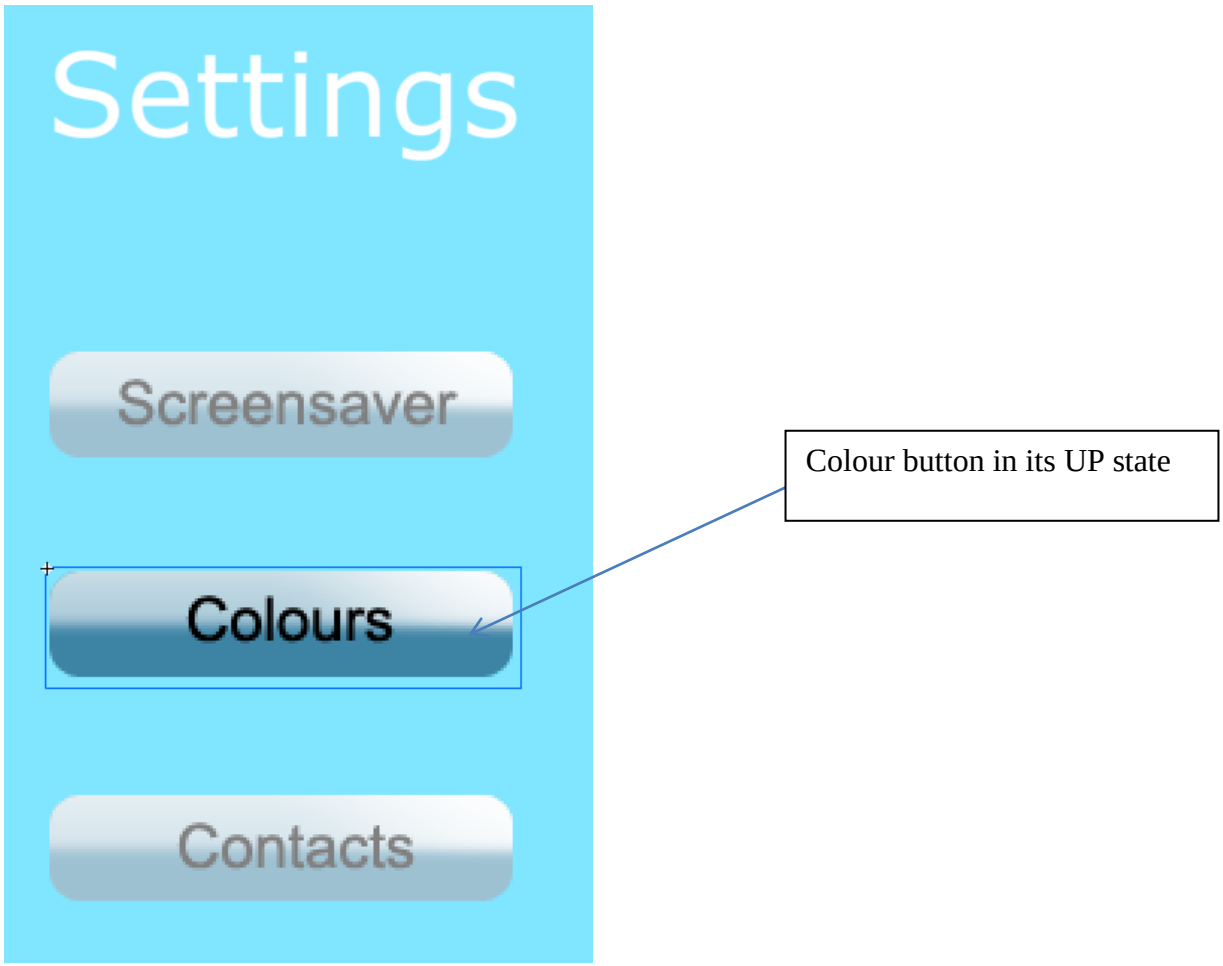


Fig.4.10: Colour button showing UP state

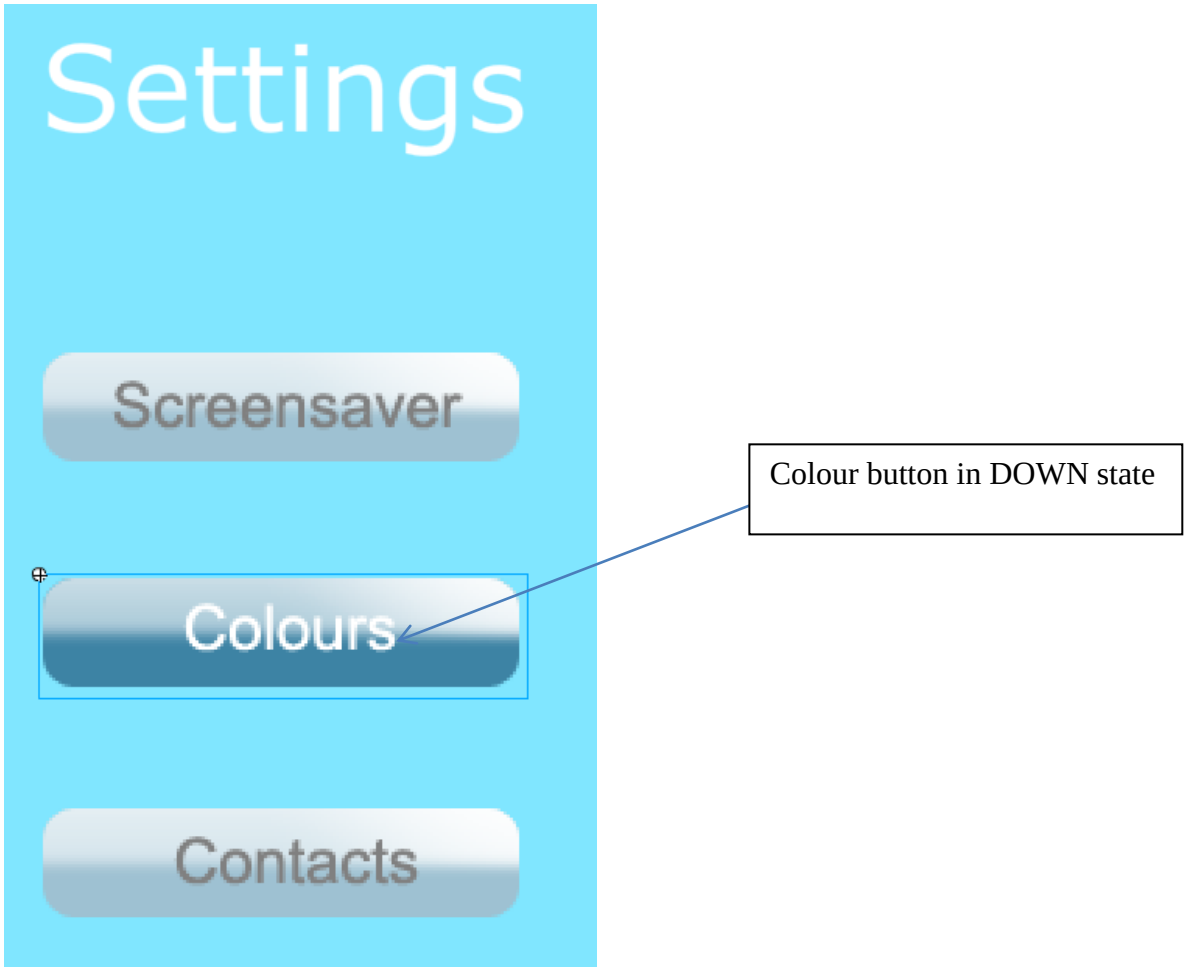


Fig.4.11: Down state of colour button

This implementation involved using the Flash colour picker component that is already contained in the Flash component library. The idea is that when a user selects a new colour from the palette, a new rectangle which fits the entire stage is drawn and the old one removed. This creates the background colour for the home screen.

```
//hex value for blue, the initial colour picker choice

var bgColor:uint = 0x0000FF;
```

The above code sets the colour of the background in hexadecimal values. I have set its initial value to blue, as this fits with the colour scheme of the main screen.

```
//fills the specified background colour

roundedRect.graphics.beginFill(bgColor);

roundedRect.graphics.lineStyle(borderSize, borderColor);

roundedRect.graphics.drawRoundRect(0,0, rectangleWidth, rectangleHeight, cornerRadius);
```

This part of the code fills the background colour in and draws the rectangle.

Once the colour picker was implemented I created a Home button which led the user back to the main screen. I used a symbol of a house to represent the home button, and created the buttons appearance similar to the other buttons created. These were decided based on the guidelines of two of Nielsen's 10 Heuristics points Consistency and standards and Match between the system and the real world.

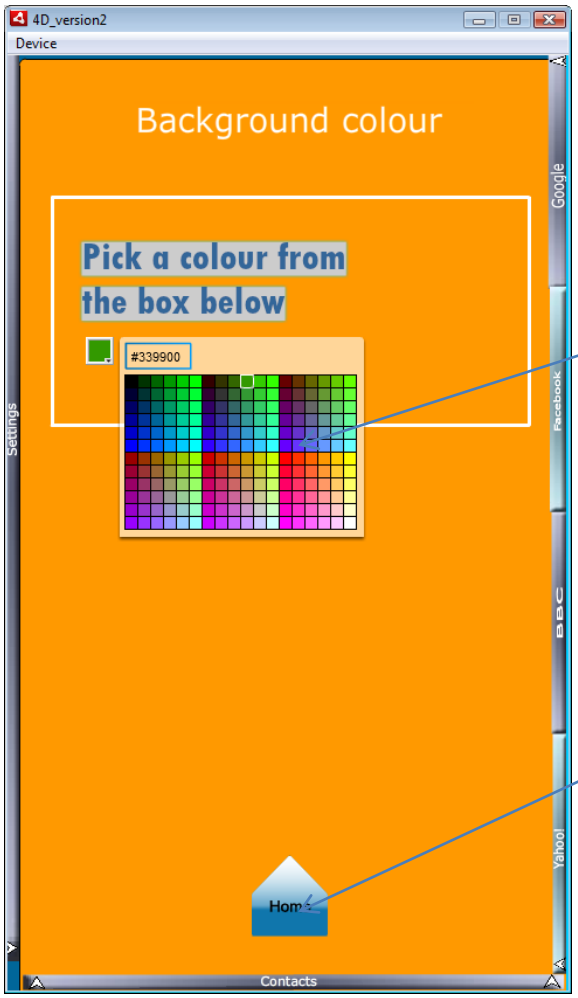
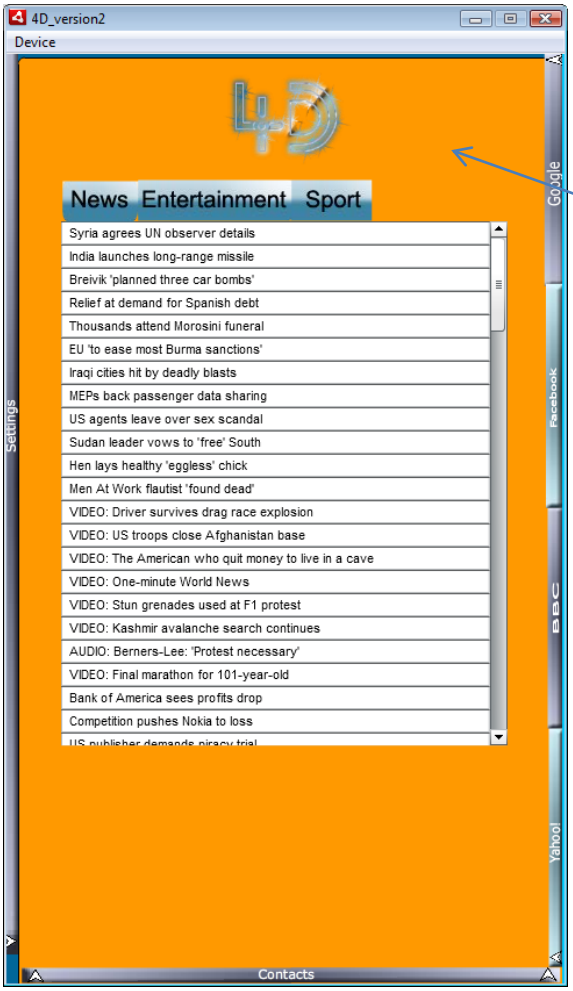


Fig.4.11: Background colour selection window



The background colour of the main screen has changed to the specified colour

Fig.4.11: Home screen with new background colour

Now that the first of the three elements of the settings panel had been complete, it was time to move on to the second part of the menu, the screen saver setting.

4.5 Screensaver settings menu

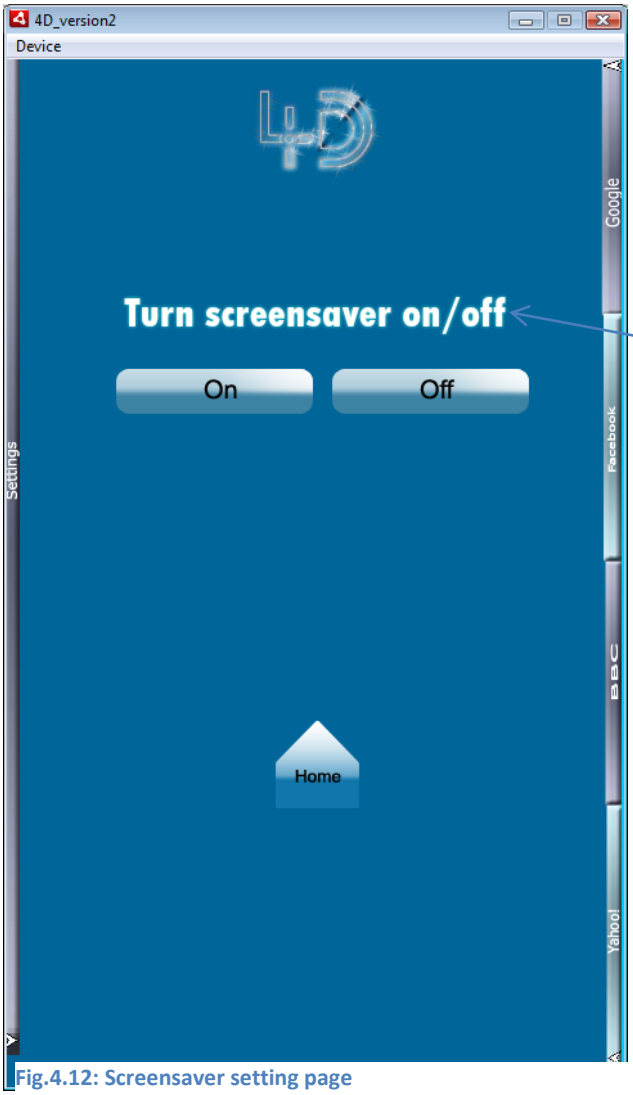


Fig.4.12: Screensaver setting page

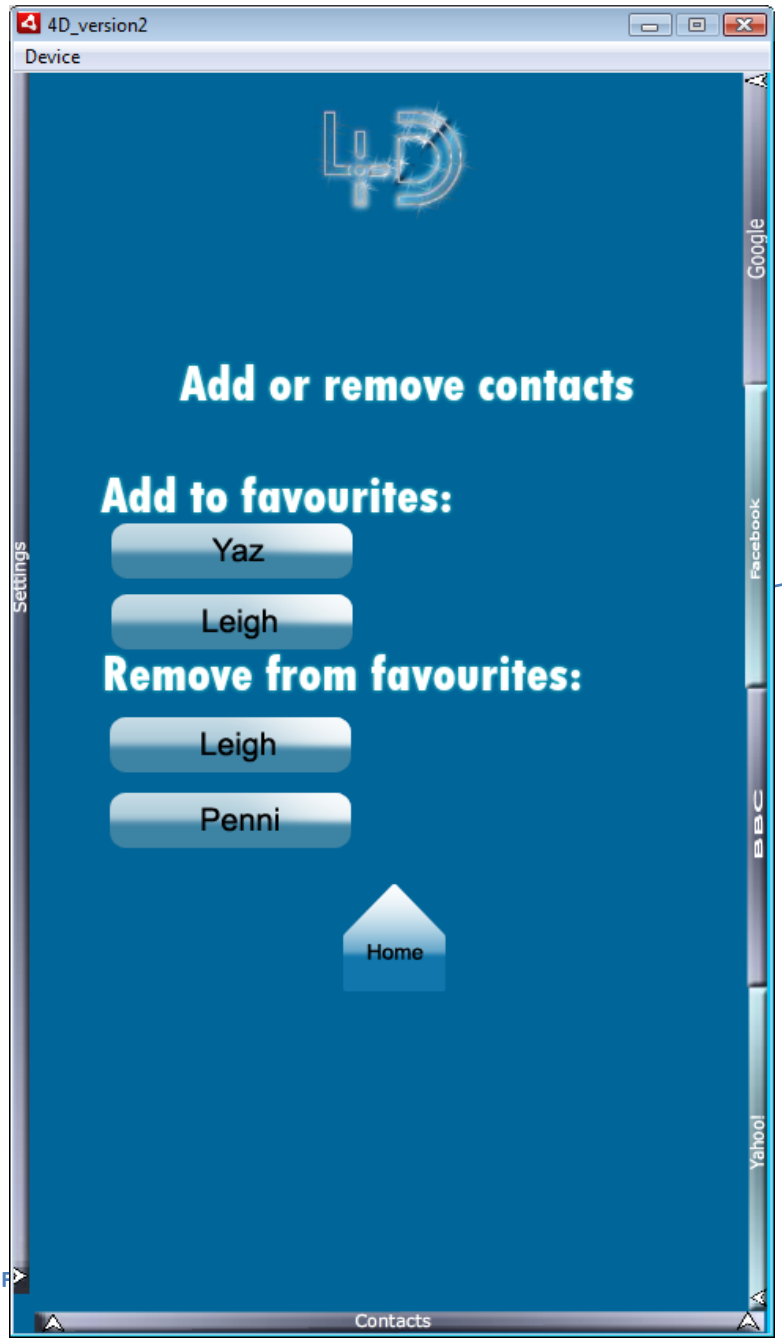
For the text I used an Arial font, both a general mobile safe font, and a specifically safe font for Android phones (Mike Mai, 2011).

The screensaver settings menu had a similar layout and design to keep consistent with the other setting menus. To create this feature I duplicated the current animating logo and created a static version of it. When the user chooses to turn off the screensaver, the two logos are swapped with each other. The same applies to when the user turns on the feature.

The last element to implement on the settings panel was the Contacts selection.

4.6 Add or remove contacts

The contacts menu in the settings panel demonstrates the ability to add or remove favourite contacts from the Contacts panel (See '4.8 Contacts panel' section). The amount of contacts that can be added and removed have been limited for demonstration purposes.



The user can select a contact to remove from here

Fig.4.14: Add or remove contacts page

The penultimate element that was left to implement was the Contacts panel, which is situated at the bottom of the main screen.

4.7 Contacts panel

The contacts pop up menu is a small panel situated at the bottom half of the interface. The purpose of the contacts menu is to allow the user to phone or text a contact stored as a favourite with the touch of a button. The menu can be accessed quickly and efficiently from the main page.

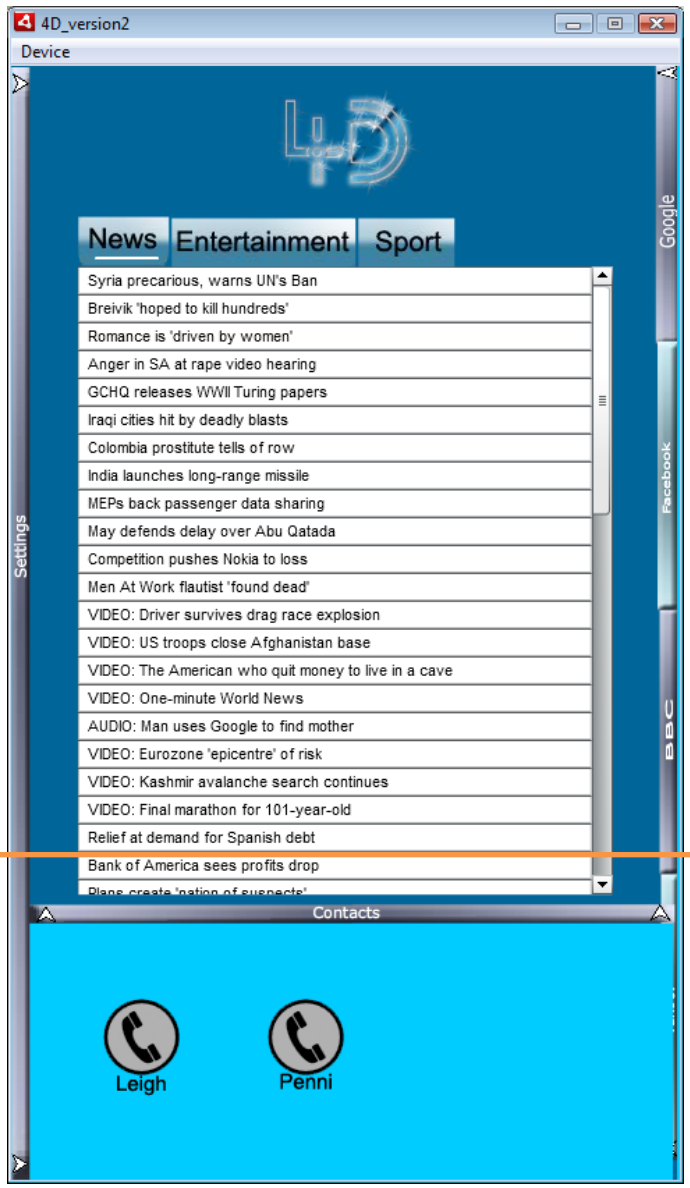


Fig.4.15: Contacts panel

To create this menu, instead of the drag and drop method used in the right handed panels, I used a simple animation that initiates when a user selects any part of the bottom panels slider. I used rounded buttons that I created in Photoshop and placed a simplistic symbol of a phone which clearly indicates the purpose of the button. I chose this symbol because it is universally understood and is often used in other applications and mobile operating systems. This fits in with Nielsen's Heuristics (1994) principle 'Match between system and real world' which expresses to "follow real world conventions".

```
function phoneLeebe(evt:MouseEvent)

{

        navigateToURL(new URLRequest("tel:07572967473"));

        gotoAndPlay(25);

        phone_btn.x = -700;

}
```

The function above is for one of the available contacts. When this function is triggered, the user is taken out of the application and into the operating system contacts options where they can text or phone the contact.

The last element to implement was a feature added after usability testing to improve efficiency when moving the panels back to their original position.

4.8 Shake to close open panels

This innovative feature allows the user to shake the mobile phone, triggering any open panels to return to their default position.

Figure 4 Fig. 4.16: Image showing the shake gesture on 4D

```
if (e.accelerationX >= 1 || e.accelerationY >= 1 ){
```

The above code measures the acceleration of both the x and y axis. The OR operator tests for both conditions. The original code also included a test for in the Z axis, so that if the device was tilted backwards or at any angle the action would occur. Because I wanted to create a 'shake' gesture, I excluded the Z axis test.

```
if (Accelerometer.isSupported)
```

The above piece of code tests whether or not the accelerometer is supported on the specific device the application is being run on. However this is only been added as a precaution as all Android devices currently have accelerometers. This may be something that changes in the future (Mario Zechner, 2011).

5.0 End of Chapter Evaluation

In this chapter evaluation I will be explaining the reasons why I decided to change certain aspects of the original designs, as well as discussing some of the challenges I faced.

5.1 Panels

The original wire frame had panels which opened to web pages or applications at every edge of the screen. The final decision was to have only the right side panels open up to web pages. Access to applications was removed after discovering that these were built into the design of the operating system, and would have required completely rebuilding the operating system.

The main challenge I faced with the panel implementation was lining up the transparent outer rectangle that acts as a placeholder for the panel object. If the position was as much as 1 pixel out the functionality may have not worked. Even after measuring the exact dimensions, other variables seemed to affect its position. I therefore had to end up using a 'trial and error' approach which led to a significant amount of time.

5.2 RSS Reader

The design of how the user updated the content was changed, from going into a menu on the settings panel, to being able to instantly flick between news feeds from the main page. This reduced the number of steps the user had to take to change the setting.

The biggest challenge associated with the RSS Reader was attempting to integrate a flick style RSS Reader, which would have enlarged the font. After successfully creating the flick style RSS Reader in a separate file using classes, it then became incompatible with my current project when trying to integrate it into the main page. Due to limited time, I decided that my time was best spent on improving other interface elements.

5.3 Contact menu

The original designs showed the contacts displayed off of a panel, and situated at the bottom of the main screen. The decision was made to place these contacts within the bottom panel, which left more room on the main screen for the RSS Reader components. This followed advice from Introducing Interactive Gestures (Dan Saffer, 2009) which discussed the importance of designing for the limited screen real estate on mobile devices.

Creating the contact menu proved less challenging than the RSS Reader and Panels. A reason for this could be that it was an element I had created towards the end of the project, and therefore I had become more familiar with the coding and software. I also managed to find a solution fairly quickly, in contrast to the RSS Reader and Panels which required a number of approaches before the best one was found.

6.0 Chapter 3: Testing

In order to find out how useable the application is I needed to conduct usability testing, as well as an expert heuristic evaluation. In this chapter I will be describing how I decided on what usability techniques to use and how I conducted them. In order to research these techniques I read a book on usability testing called Usability Testing Essentials (Carol M. Barnum, 2010). It enabled me to enhance my knowledge in this field and also help me decide on the best techniques to use for my specific application.

6.1 Understanding users

Jakob Nielsen did a study to address the common feeling that web users have now become smarter. In his tests he found that experienced web users are now better at controlling input peripherals such as mice and keyboards, more confident with clicking and not as afraid they will break something, have a better understanding of how to use search engines, and faster at doing things they do regularly on websites they visit often. However, a key finding in this experiment was that users felt confused when entering websites they were unfamiliar with, lacking an understanding of its structure and menu options (Barnum, 2010).

As 4D features a different type of interface, Nielsen's findings, although based on websites viewed on desktops, could reflect on user's feelings when using 4D for the first time. However, the interface has been built intuitively, without the need for many instructions, and basis most of its elements is based on those which have been used in other applications and operating systems.

When users want to achieve something quickly, they look for trigger words which match their end goal (Barnum, 2010). They do not want to read through lots of text to find one thing that interest them. 4D's RSS Reader addresses this by outlining only a small section of a story in a list format, which they can then click to access a longer description.

First impressions are critical to user's perception of how difficult or easy a website is to use. One study showed that 80% of web users spend only a few seconds on a page before deciding whether the site is worth exploring or not (Barnum, 2010).

6.2 Expert review on 4D using Nielsen's Heuristics

An expert heuristic evaluation (or expert review) was carried out on the application using Nielsen's 10 Heuristics. Studies suggest that usability experts find twice as many minor errors as major problems. It is therefore advised to bring the best of both worlds together, and conduct both usability testing and an expert evaluation (Burnum, 2010). I conducted an expert evaluation first to effectively 'clean up' the interface first, to get rid of any problems that might interfere with the user's experience. This would help the user focus more on the experience and less on any distractions.

Nielsen's Heuristics	Issue	Solution
Visibility of system status	There is a lack of indication of what page a user is on at a current time	Provide clear titles that say exactly where the user is at that time
	When selecting either on or off in the shake feature menu, there is no status as to the current state	Show the current state of the feature by indication of the down state of the button
	When the user shakes the device for updates there is no feedback as to whether the shake has been recognised	Create a spinning logo that plays when the shake has been recognised
	Users do not know which panel corresponds to which web page	Provide labels on the panels stating the corresponding web page
Match between the system and the real world	User will not know which gesture to use to access the applications using the panels	Use the press and drag technique currently used on both iOS and Android OS to pull notification panels
	Customise panel but not be immediately obvious to its function	Use a symbol which represents a tool i.e. spanner and clearly label the page
	Using only words to represent the application	Provide a symbol or logo of the application next to

	name on the panels slider might take time for the user to identify it	the written text
User control and freedom	In the customize section, when users select to change a setting they may make a mistake	Include a confirmation dialogue box on settings which are difficult to reverse back to their previous state
Consistency and standards	Menu buttons are not aligned properly	Ensure all menu buttons are aligned properly
	Swipes appear to trigger different responses depending on the page you are on	Ensure swipe gestures stay consistent throughout i.e. swipe left returns to previous page
	Buttons sizes seem disproportionate	Ensure all button sizes are of a similar size
Recognition rather than recall	Panels disappear from view when accessing an application	Make sure all panels are accessible at all times apart from when in the customise section to minimise the user having to backtrack
Help and documentation	No help section available	Include simple and clear help section

Reference: Jakob Nielsen, 2005, *Ten Usability Heuristics*, [online], Available at: http://www.useit.com/papers/heuristic/heuristic_list.html

6.3 Usability testing

Conducting a small study

Conducting small studies with around 5 or 6 participants can be beneficial in a number of ways. It is especially useful when time and costs are limited. My decision to conduct a small study was based on weighing up the advantages and disadvantages of both. As time and budget was a factor and my application was not extremely complex, conducting a small scale study was appropriate.

In order to conduct this study I needed to establish a user profile from a subset of the total user population. My user profile would be those that are regular users of smart phones, regular being at least once a day. They must also use their smart phone to browse the internet at least once a week. The first part of this study required me to create a set of tasks for the user to perform. There would be set goals to ensure the user does not get lost or stray from the objective they are set. This is important in order to obtain clear results and patterns of usage.

Questionnaire

The second part of the test involves the user answering a number of questions based on Nielsen's Principles that reflects the tasks they have just performed. The format is in the form of a questionnaire using a Likert scale, "a psychometric scale commonly involved in research that employs questionnaires" (Kendra Cherry, 2012). The five level rating scale ranges from strongly disagree to strongly agree. The users must put a tick in one of the checkboxes on the scale, and they are free to go back and complete the tasks again if they wish. The sample questionnaire including the set tasks can be found in APPENDIX J.

Observation

After receiving the questionnaires back, I was able to get an idea of what users liked and didn't like about the interface, yet it didn't allow me to obtain specific details about any particular element. It was here that I decided to also conduct an observation, where I could communicate with the tester and write down notes as they used the application. I found this method even more useful than the questionnaire, as I was able to see first-hand what users were struggling with.

Results

After the questionnaires had been returned and the heuristic evaluation and observation was complete, the results needed to be analysed to discover which ways the application needed to be improved to enhance usability. The table below summarises the key issues that needed improvement, together with the solutions.

Issue	Solution
It isn't clear what the customization panel is, and how to open it	Label added to slider panel, and arrow prompts indicating users to slide to open
It is not clear what web pages are on the right handed panels until you open them	Label added to each slider on the right handed panels, two arrow prompts added, one on panel 1 and 1 on panel 4
When on the customization panel, there is no indication that you are on that particular panel	Title added to customization page
The grey panels do not go well with the overall theme, and look dull.	Add aqua blue colour to all panels
The shake features purpose is unnecessary and should be used for another feature	Shake features functionality changed so that it closes right handed panels
Right handed panels are easy to open, but fiddly to slide back into position	Shake feature utilised to aid this process. Shaking the device now closes any open right handed panel
It is unclear what the 'Shake' option does in the customization menu	'Shake' title now changed to 'Screensaver', which leads to another page where users can turn on or off the screensaver

The shake features purpose is unnecessary and should be used for another feature	The previous feedback from the spinning logo has now been changed to a screensaver
It takes an unnecessary amount of steps to change the RSS Reader content	Instead of changing the RSS Reader content in the settings panel, it now uses tabs situated on top of the reader.
There is nowhere to add or remove contacts	The RSS Reader setting in the settings panel has now been changed to an Add/remove contacts menu

7.0 Chapter 4: Discussion and Evaluation

At the end of each chapter I have included an evaluation of that specific chapter. In this section I will be summarising those chapters, explaining what I have achieved and how I went about it. A small section on management is also included.

7.1 Chapter 1: Requirements and Design

In this stage I was able to apply the knowledge I had gained from research into HCI principles and mobile interface design. I created a set of deliverables that both represented the functionality and visual elements of the application in the form of a Requirements Specification, wire framing, use case diagrams and task analysis. I used a dedicated UML tool to create clear and consistent diagrams, together with paper based techniques to draw simple, yet effective sketches to convey my designs. As the stages progressed, I had to iterate the design process to address limitations and design changes that were made at a later date.

7.2 Chapter 2: Implementation

In the implementation stage I broke down the designs into manageable segments. I then took a systematic approach by dividing these sections up into 3 stages of implementation: functionality, testing, and visual design. For example, when implementing the panels, I started off with a very simple blank object with no visual design. This was created very quickly, enough so that functionality could be tested. I then went on to writing the code for the object, before testing its functionality was satisfactory. The final stage was to overlay the aesthetic elements of the object. The reasons I took this approach was that it was more manageable, logical and reduced time being wasted.

I was faced a number of challenges in the implementation stage. There were programming elements that I had not come across before, such as using the devices accelerometer to produce the shake feature, and incorporating touch screen gestures. I was able to tackle these by reading through books and online material to learn how these work. There were also some elements which I could not fully implement to the degree they were initially specified in the design stage. This was particularly true for the RSS Reader, where I could not use the style I initially planned so the font was a little too small. This however, was a key learning point as I now know how best to structure the programming in future work.

7.3 Chapter 4: Testing

As well as testing the functionality after each element was incorporated as described in the 'Implementation' chapter, it was also very important for me to conduct thorough usability testing and evaluation by using specific techniques that were suitable for my interface. To ensure that I used the best methods and conducted these in the correct way I read a book on usability called Usability Testing Essentials (Carol M. Barnum, 2010).

After getting an idea of the different methods that can be used to test usability I decided to use three of them; a questionnaire that included the tester completing a set of tasks using the application, and then answering a set of multiple choice questions using a Likert scale; an heuristic evaluation using Nielsen's 10 Principles; and finally an observation where the user would explain their thoughts in an informal session where notes were taken. The reason I chose these 3 techniques was because they each gave me a different level of evaluation. The heuristic evaluation was more technical and was heavily based on traditional heuristic guidelines, whilst the questionnaire was less technical and was easier to spot trends. The observation was informal and was less based on heuristics, yet still allowed me to obtain more specific detail about what problems users faced. These 3 techniques combined to make a comprehensive usability testing plan.

Overall I found the testing stage very helpful in enabling me to see the perspective of a possible end user of the application, and what improvements they would suggest. It also gave me an insight into what the users liked about the interface, as these could also be built upon.

7.4 Management

Weekly reports

To enable me to plan my work schedule out effectively I made use of the weekly reports where I was able to summarize what I had done for that week, and plan for the following week. If I was unable to succeed in completing a particular aspect of work in any given week, I would evaluate why this was and what I plan to do about

it. I also managed time by comparing how long I originally planned for something to complete, with how long it actually took to take. This gave me an idea on how long a similar task would take in the future.

Report writing

In terms of writing my report, I chose to start writing it up in rough as I went along, rather than waiting until everything had been implemented and tested. I found this method useful because events were fresh in my memory, and building up a structure was easier than leaving the report to the last minute. I planned to write a rough amount of words each week, but focused on content rather than quantity of words.

Usability testing and evaluation

I had to be careful when considering what methods I was going to use when conducting usability testing and evaluation. Due to time limitations, I conducted a small study involving a select group of individuals. An advantage of using smaller groups is that I was able to use 'observation' as a technique, which led to more focused results. A disadvantage of using smaller groups is that I couldn't get a true representation of a wider audience.

References

To ensure that I kept track of the material I had used for my project that would later be used to create my reference list and bibliography, I saved all web pages in my favourites and kept a written log of the books and journals I had read. I also created a Word document with examples of how to reference using Harvard Referencing techniques as I quick guide.

7.5 Future development of 4D

There are a number of areas that I feel could be improved for possible future development. Below I have outlined some of them:

- The RSS Reader style could be changed to a flick menu, which would improve scrolling and enlarge the font
- The RSS Readers capability could be expanded to allow the user to view a larger article when a specific story is selected
- As well as changing the background colour, the user could have the ability to choose images to have as backgrounds
- A music player could be integrated and make further use of the shake feature

This brings my report to a close. Thank you for reading. Below you can find a list of references and a bibliography, as well as the appendices.

8.0 References

- Saffer, D.S., 2009, *Introducing Interactive Gesture*, Sebastopal, CA
- Hoober, S, Berkman, E, 2011, *Designing Mobile Interfaces*, Beijing Cambridge Farnham Köln Sebastopol Tokyo
- Trevor Mogg, 2012, *Mobile devices to outnumber people on planet this year*, [online], Available at: <http://www.digitaltrends.com/mobile/mobile-devices-to-outnumber-people-on-planet-this-year/>
- Aaron Smith, 2011, *One quarter of smartphone owners use their phone for most of their online browsing*, [online], Available at: <http://pewinternet.org/~/media//Files/Reports/2011/PIP_Smartphones.pdf>
- Jakob Nielsen, 2005, *Ten Usability Heuristics*, [online], Available at: http://www.useit.com/papers/heuristic/heuristic_list.html
  Mike Mai, 2011, *Choosing Fonts for Your Mobile Website*,[Online], Available at: <http://www.bluetrainmobile.com/blog/choosing-fonts-for-your-mobile-website/>
  Jakob Nielsen, 2011, *Mobile Usability Update*, [Online], Available at: <http://www.useit.com/alertbox/mobile-usability.html>
- Shneiderman, B. (2004). *Designing the User Interface: Strategies for Effective*
  Human-Computer Interaction .Addison-Wesley Publishers, Reading, MA
- Daniel Rubino, 2012, *Study: Apps not that important to smartphone users*,[Online], Available at: <http://www.wpcentral.com/study-end-apps-not-important-smartphone-users>
- Zechner M, 2011, *Reading the accelerometer state chap. 4 pg.141*, New York, NY

9.0 Bibliography

- Project leader: Linus Tolke, *ArgoUML*, UML Modelling Tool, Version 0.28.1, Website: <http://argouml.tigris.org>
- Adam Khoury, 2009, *AS3 ColorPicker Component Tutorial: Color choosing*, [Online], Available at: <http://www.developphp.com/view.php?tid=274>
- Username: RapsFan, 2011, *Tablet OS SDK for Adobe AIR*, [Online], Available at: <http://supportforums.blackberry.com/t5/Tablet-OS-SDK-for-Adobe-AIR/Accelerometer-shake-trigger-in-Flash/td-p/748727>
- Username: Mirelatm, 2010, *StageWebView - HTML Support in AIR for Android*, [Online], Available at: <http://forums.adobe.com/thread/750186>
- Support Forums, 2012, [Online] Available at: <http://www.supportforums.net/>
- Adam Khoury, 2009, *ColorPicker*, [Online], Available at: <http://www.developphp.com/view.php?tid=274>
- *How to write a software requirements specification (SRS) document*, [Online], Available at: <http://www.jaysonjc.com/programming/how-to-write-a-software-requirements-specification-srs-document.html>
- *Mobile App Step-by-step #5 - Publishing a mobile application in Flash*, [Online], Available at:< http://tv.adobe.com/watch/mobile-applications-101/publishing-a-mobile-application-in-flash/>
- Russ Tarleton, 2011, *Detecting shakes on a Flash 10.1 enabled Android Device using ActionScript*, [Online], Available at: <http://www.russtarleton.com/>
- Christian Cantrell, *URI Handlers in AIR for Android: Phone Calls, Email, Text Messages, Maps, Market, and URLs*, [Online], Available at: <http://blogs.adobe.com/cantrell/archives/2010/11/uri-handlers-in-air-for-android-phone-calls-email-text-messages-maps-and-urls.html>
- Username: teachucompinc, 2011, *Word 2010 Tutorial Creating a Table of Contents Microsoft Training Lesson 19*, [Online], Available at: <http://www.youtube.com/watch?v=zt4AITbQmpU>
- Lee Brimelow, 2010, *Optimizing Drag and Drop for Mobile Application*,[Online], Available at: http://www.leebrimelow.com/?p=2537

Appendices

APPENDIX A – Nielsen's 10 Heuristics

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

APPENDIX B – Shneiderman's '8 Golden Rules of Interface Design'

1 Strive for consistency.
Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent commands should be employed throughout.

2 Enable frequent users to use shortcuts.
As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. Abbreviations, function keys, hidden commands, and macro facilities are very helpful to an expert user.

3 Offer informative feedback.
For every operator action, there should be some system feedback. For frequent and minor actions, the response can be modest, while for infrequent and major actions, the response should be more substantial.

4 Design dialog to yield closure.
Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives the operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions.

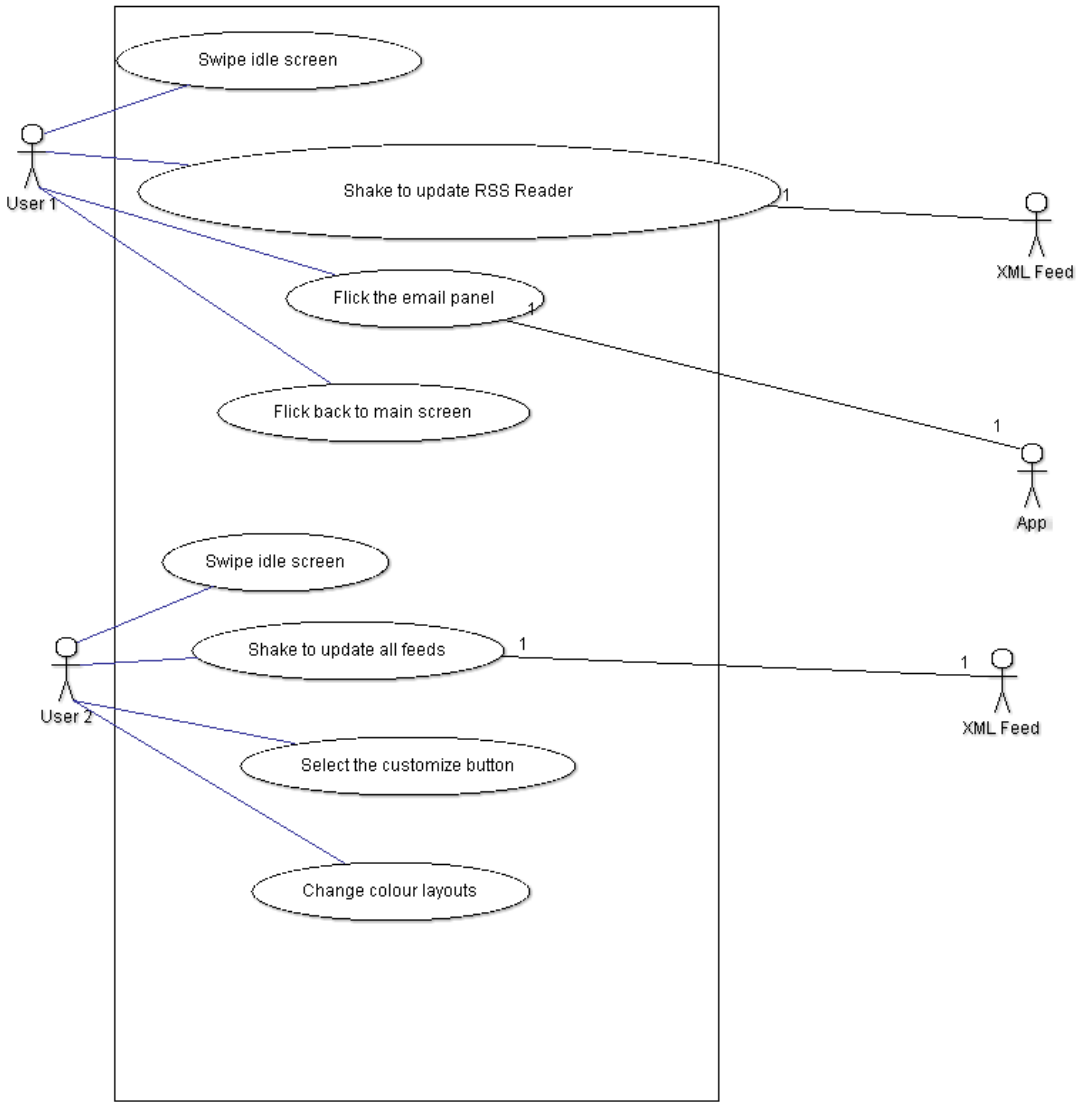
5 Offer simple error handling.
As much as possible, design the system so the user cannot make a serious error. If an error is made, the system should be able to detect the error and offer simple, comprehensible mechanisms for handling the error.

6 Permit easy reversal of actions.
This feature relieves anxiety, since the user knows that errors can be undone; it thus encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data entry, or a complete group of actions.

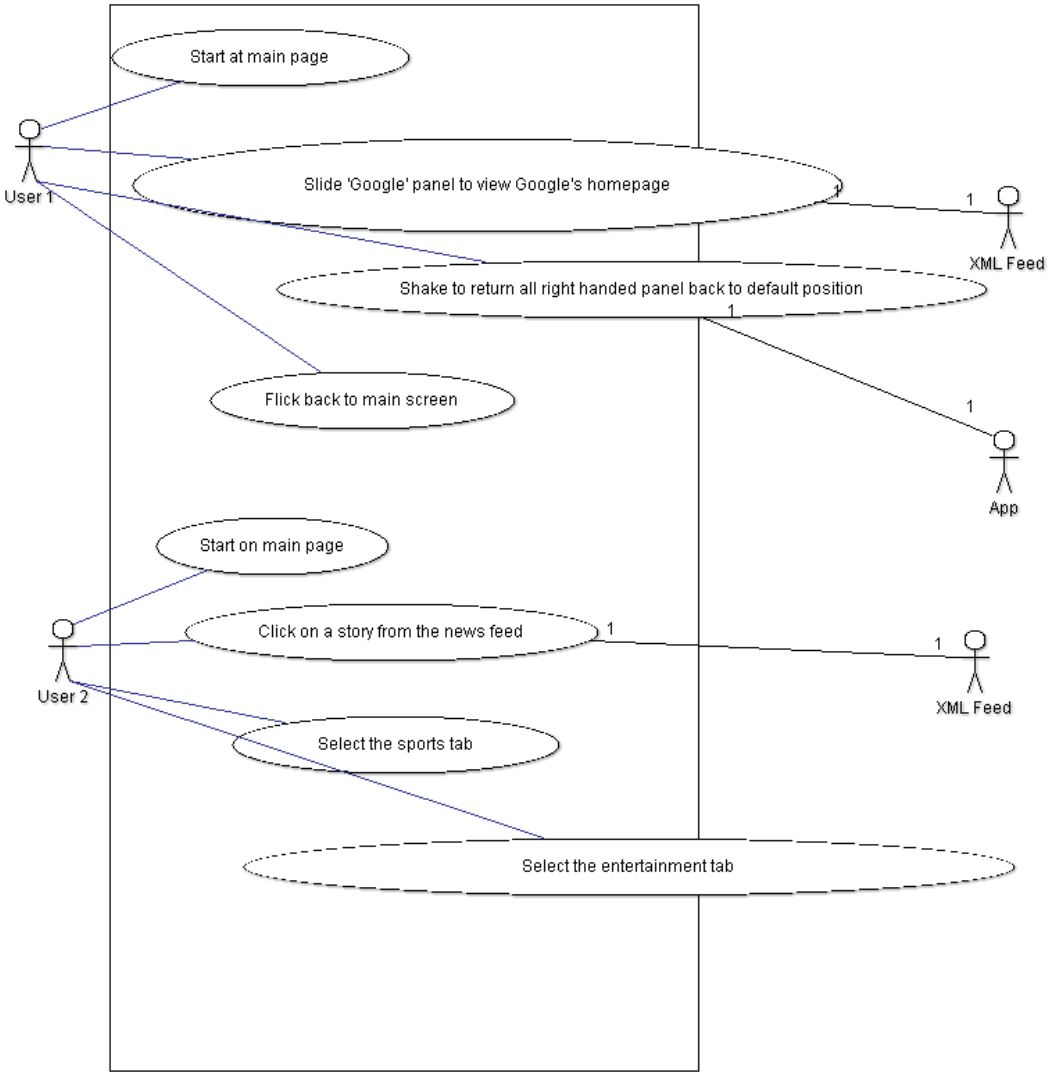
7 Support internal locus of control.
Experienced operators strongly desire the sense that they are in charge of the system and that the system responds to their actions. Design the system to make users the initiators of actions rather than the responders.

8 Reduce short-term memory load.
The limitation of human information processing in short-term memory requires that displays be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequences of actions.

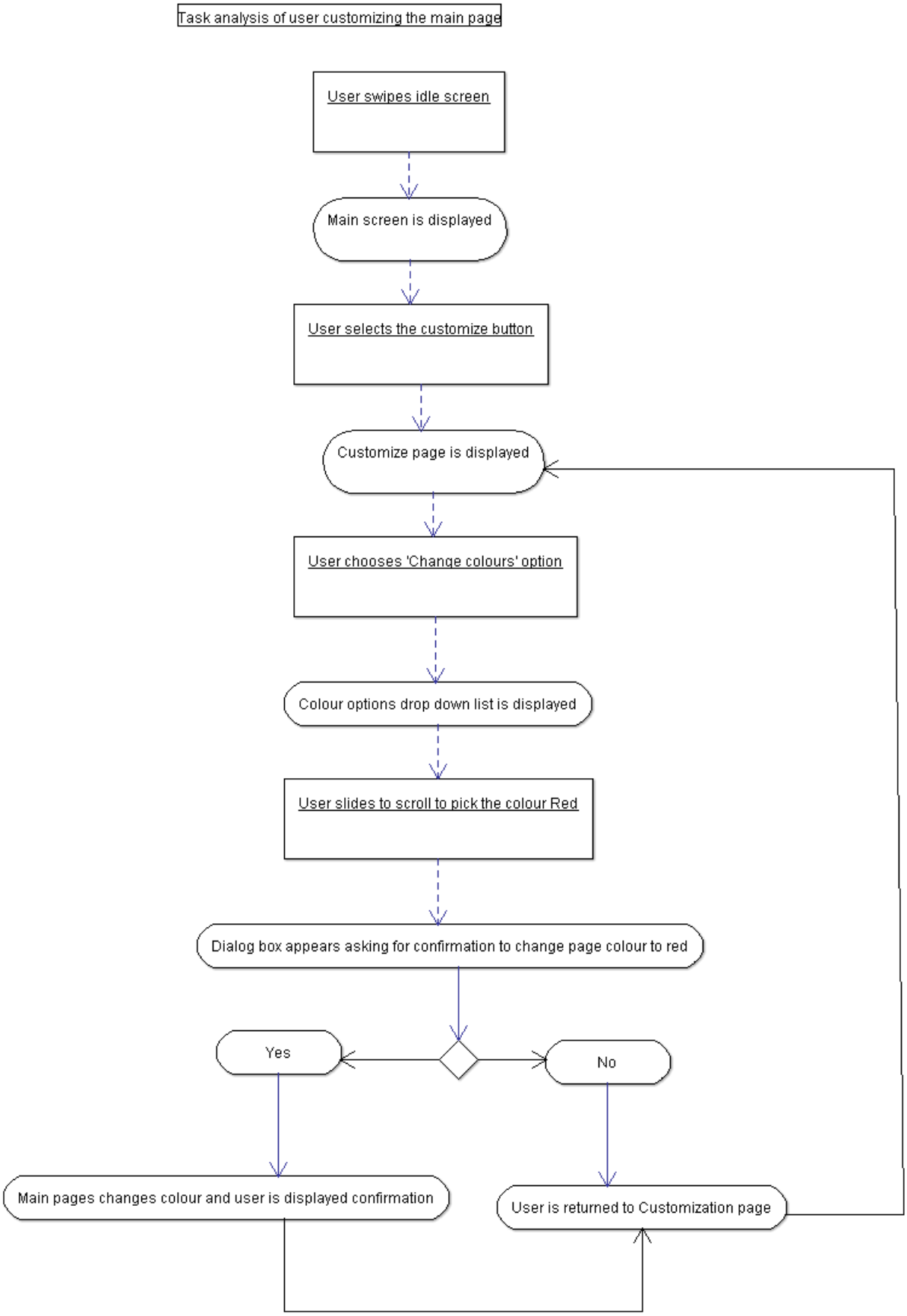
APPENDIX C- Use Case Diagram Version 1



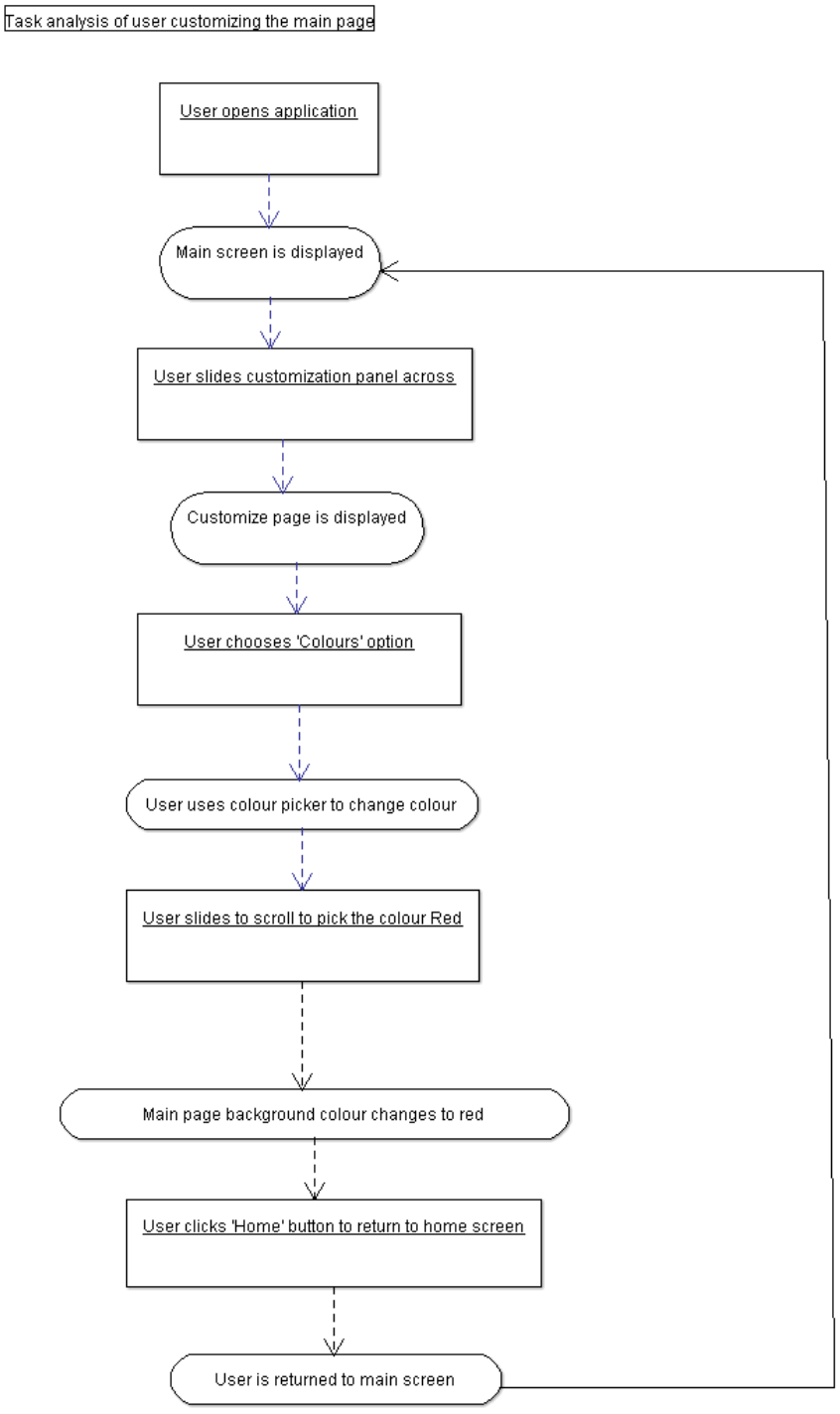
APPENDIX D- Updated Use Case Diagram



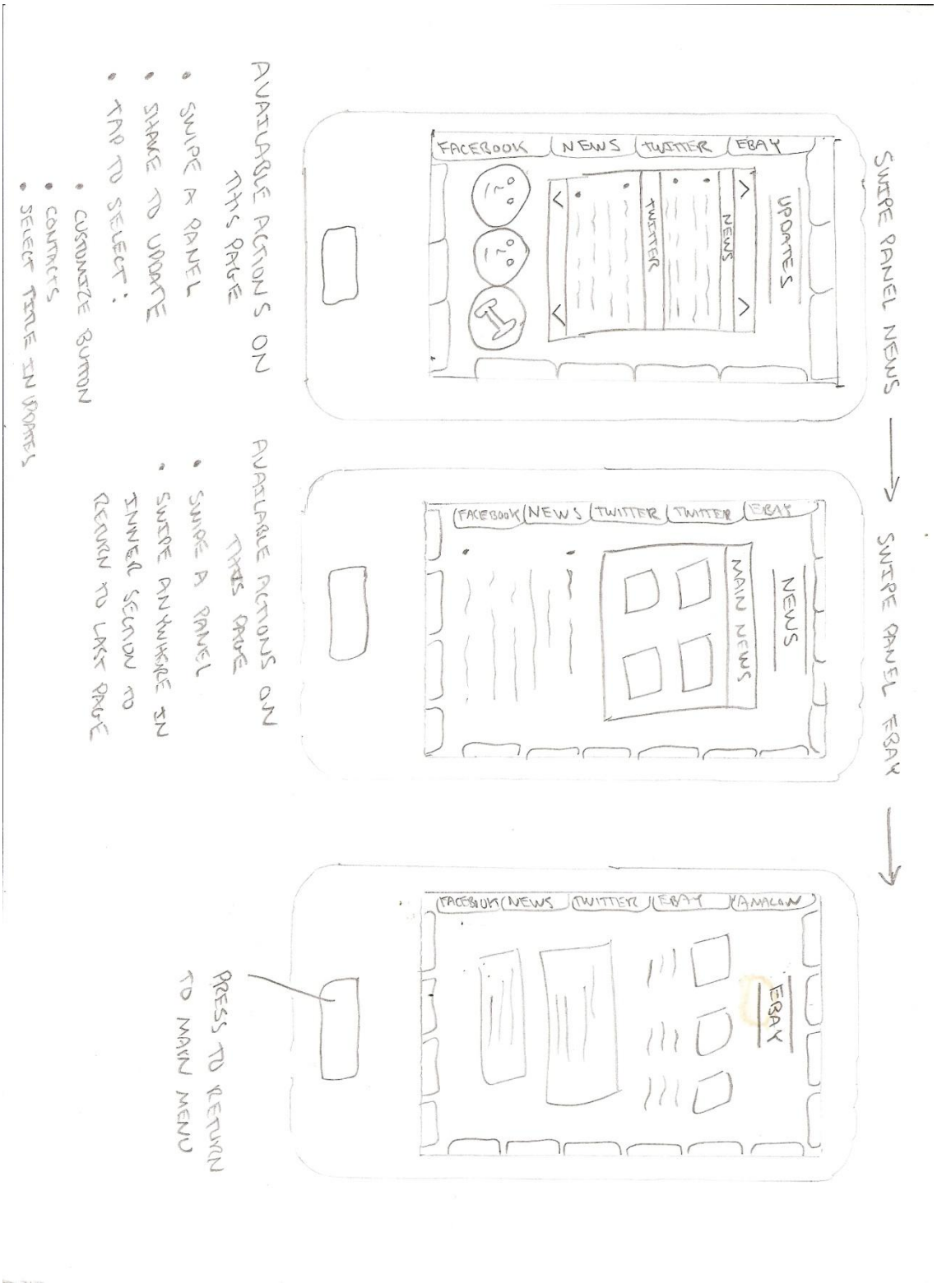
APPENDIX E – Task Analysis Diagram 1

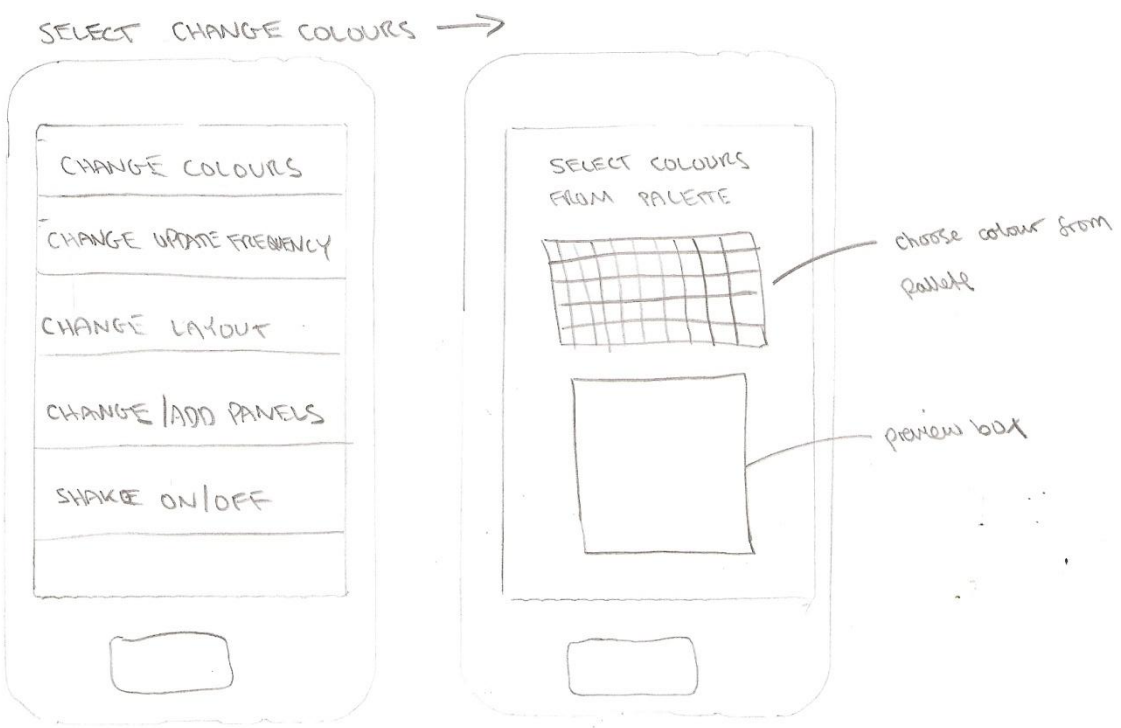


APPENDIX F – Task Analysis Diagram 2



APPENDIX G – Initial wire frames





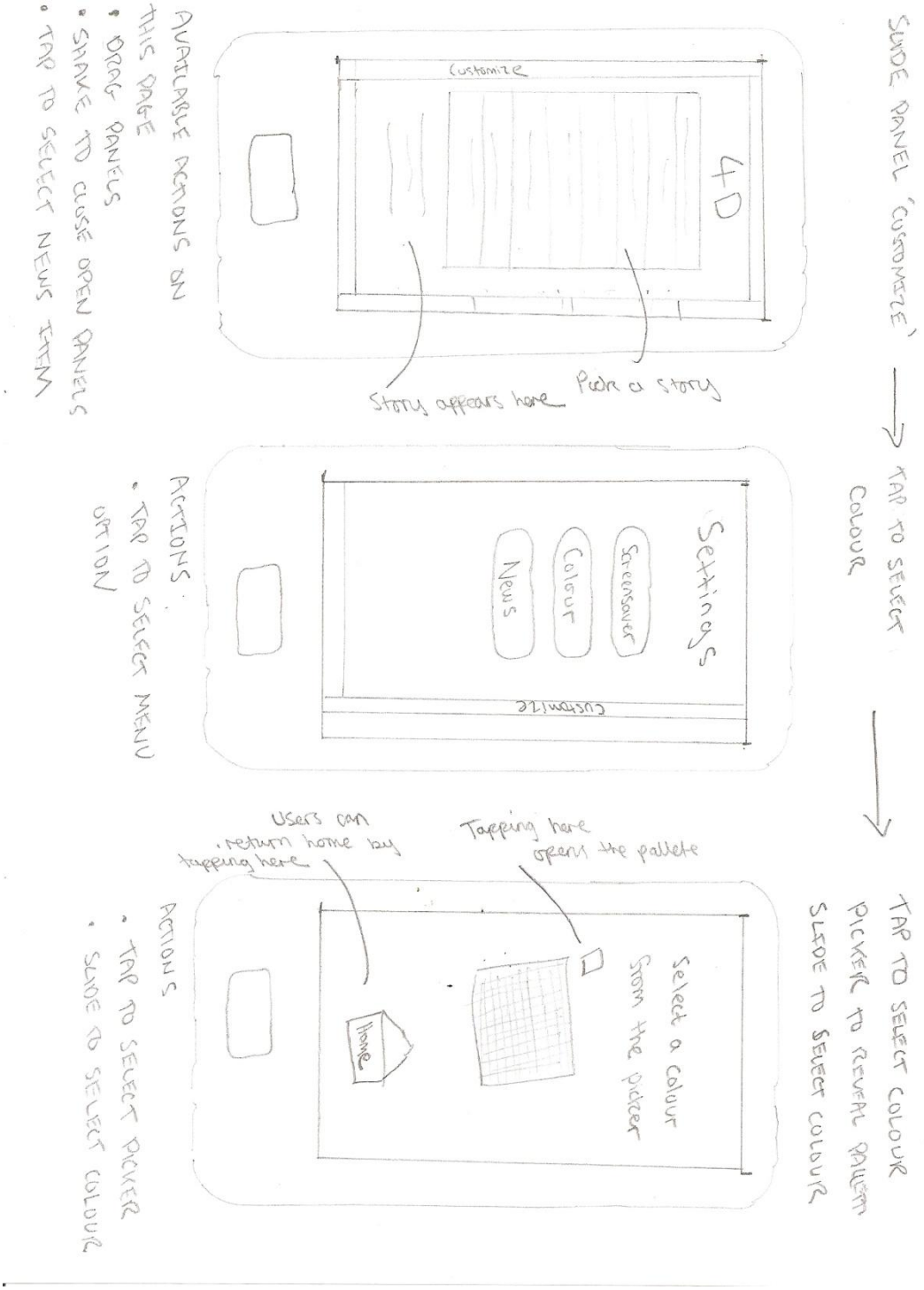
Actions on this page

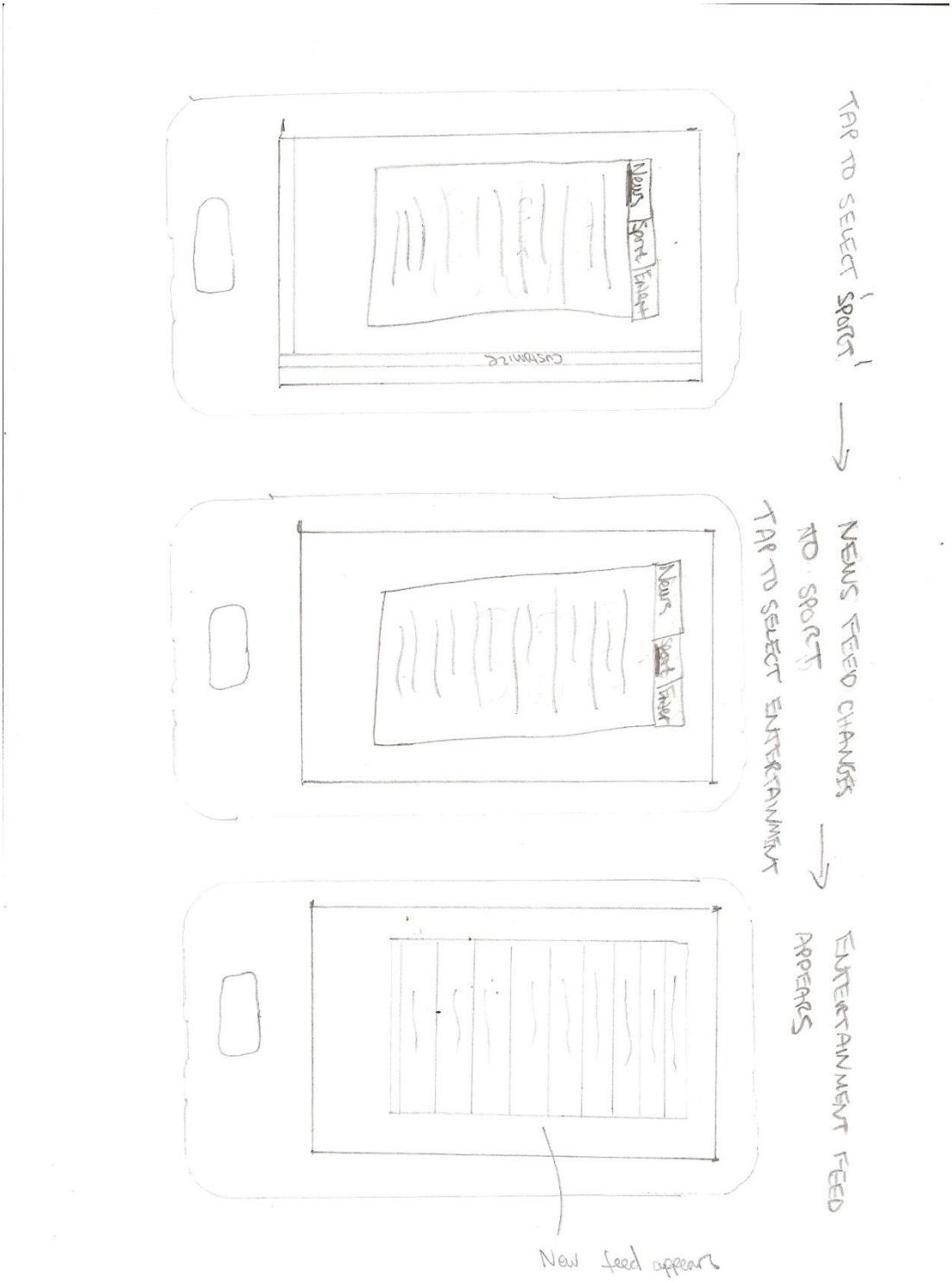
- Select from a number of customization options
- Tap to select

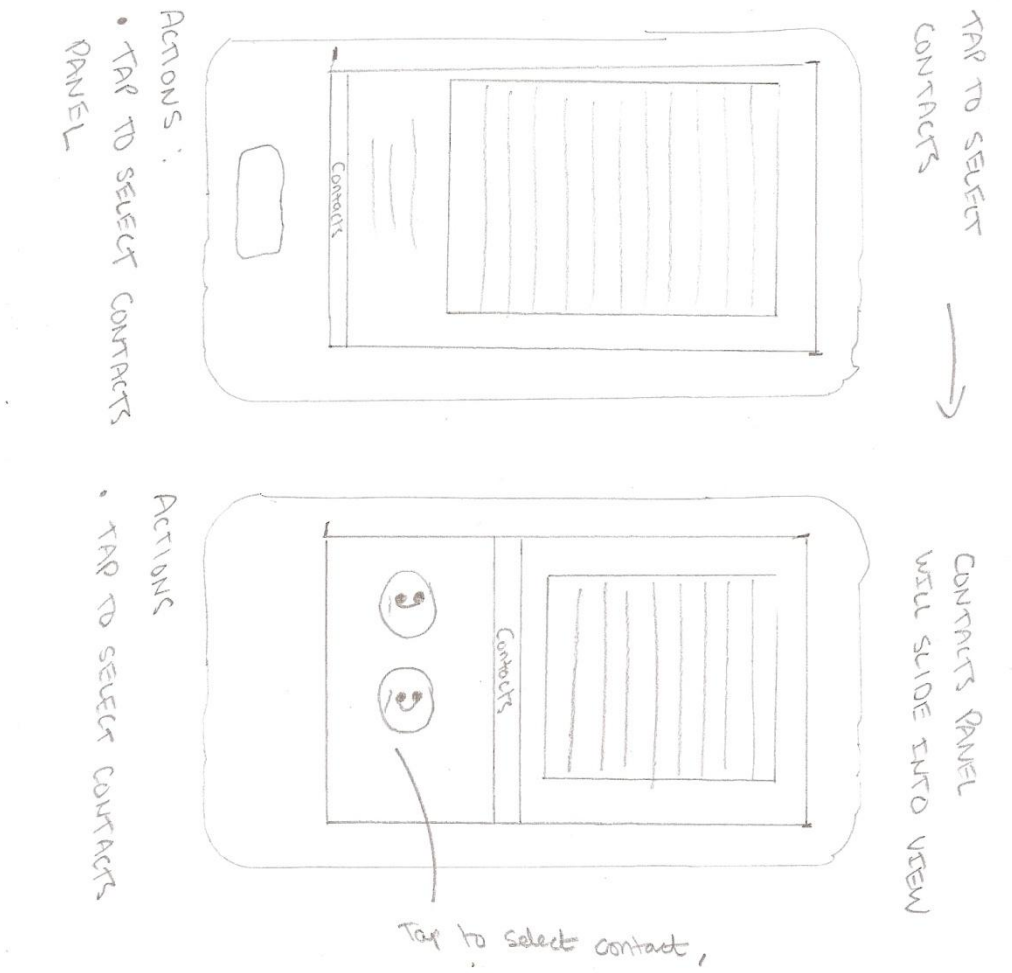
Actions on this page

- Tap to select / slide to select a colour from the pallete.
- Swipe anywhere to the left to return to the customisation page

APPENDIX H – New wire frames




TAP TO SELECT SPORT
NEWS FEED CHANGES TO SPORT
TAP TO SELECT ENTERTAINMENT
ENTERTAINMENT FEED APPEARS
News
Sport
Entert
Customize
New feed appears


TAP TO SELECT CONTACTS
Contacts
ACTIONS:
• TAP TO SELECT CONTACTS PANEL
CONTACTS PANEL WILL SLIDE INTO VIEW
Contacts
ACTIONS
• TAP TO SELECT CONTACTS
Tap to select contact,

APPENDIX I – Requirements Specification Document

SOFTWARE REQUIREMENTS SPECIFICATION FOR 4D

1. Introduction

1.1 Purpose

The Purpose of this Software Requirements Specification is to provide the technical, functional and non-functional features of the 4D software. The primary purpose of 4D is to create an efficient interface that allows users to effectively navigate to web pages and view news feeds quickly and effortlessly on a touch screen mobile device. Other features include customizable options and the inclusion of other gesture based interaction. The intention would be that this could be used as an update to the Android operating systems user interface.

1.2 Scope

The scope of this project is to design and implement an interactive user interface with the 4 elements being:

- the main graphical user interface including panels
- RSS reader
- customization section
- touch screen gestures

2. Systems overview

2.1 Application perspective

In a similar way as seen on the notifications pull down tray on both iOS and Android, the 4D interface will allow users to drag a panel to a desired position on the screen. There will be many panels surrounding the screen which can be used to access web pages and other content. Users can access another menu that allows them to customize the interface in several ways.

2.2 Operating environment

- Windows 7 64 bit OS for development
- Android Mobile Operating System for deployment.

2.3 User Characteristics

- Developer
  - Skills in coding using ActionScript 3.0
  - Working with Adobe Flash CS5.5
  - Knowledge in designing for touch screen devices
- End user
  - Users who depend/desire fast navigation
  - Learning curve is fast as interaction is intuitive

3. Functional requirements

3.1 Functional requirement 1: Navigation panels and buttons

Description

There will be a number of panels that surround the outer edge of the main screen. The user will be able to use the gestures of drag and drop, tap to select and shake, to interact with these panels. Buttons will also be used where swiping gestures are not appropriate. These more conventional methods will allow the user to return to the main screen, access new menus and select new options.

Rationale

On the current Android OS, to access a web page the user must first open up the web browser. To then open a new page they must navigate through 3 steps. To close the page they then must go through at least 2 steps. 4D allows users to access web pages straight from the home screen, simply by dragging a panel across the page to a desired position. Closing the web page requires nothing more than a quick shake of the device.

Requirements

The end user is required to have the following:

- An Android phone running the Android Mobile Operating System

The end user would benefit from having the following:

- Internet connection

3.2 Functional Requirement 2: RSS Reader

Description

The RSS Reader will extract XML files to provide information such as news and entertainment feeds.

Rationale

The RSS reader allows users to view the latest news stories at a quick glance. They can then click on a news story to open up a longer descriptions at the bottom of the main screen.

Requirements

The end user is required to have the following:

- An Android phone running the Android Mobile Operating System
- Internet connection

3.3 Functional Requirement 3: Customization menu

Description

This menu which is accessed via a the left handed panel takes the user to a new menu which allows the user to customize the user interface in several ways. These include:

- Changing the home screens background colour.
- Turning off or on the screensaver
- Adding and removing contacts

Rationale

The purpose behind the customize menu is to provide the user with increased freedom and flexibility to tailor the interface and functionality to which they desire.

Requirements

The end user is required to have the following:

- An Android phone running the Android Mobile Operating System

The end user would benefit from having the following:

- Internet connection

4. Software/hardware requirements

- Developer
    - Adobe Flash CS5.5 with minimum system requirements to support it
    - Google Android SDK 1.0
    - Android phone with latest OS for testing and deployment
    - XML for use in RSS Reader
- End user

    - An Android phone running the Android Mobile Operating System
    - Internet connection (required for RSS Reader and web pages)

5. Non-functional requirements

- Fast internet connection (not essential but would enhance user experience and performance of the RSS Reader)
- Smartphones with good battery life (as multiple applications may be open at the same time using the panel interface)
- Good 3G connectivity with fair data usage (not essential but user experience could be greatly increased especially with the RSS Reader)
- Smartphone with effective touch screen technology (this would be beneficial as the interface is fully dependent on touch screen gestures of various types)
- Screen size resolution adjustments for multiple sized handsets

APPENDIX J – Questionnaire sample

Usability Questionnaire
Based on Neilson's Heuristics

Tasks

Complete the below tasks using 4D, then answer the following questions. You can complete the tasks as many times as you like.

Task 1: Updating the RSS Reader

- Open the 4D application
- Turn on the shake feature by opening the left handed panel and choosing the Shake button. You will then be presented with a new screen. Press the button marked 'On'.
- Shake the mobile device; you should see the 4D logo spin briefly. This indicates that the RSS Reader is updating. Note that you may see no changes in the RSS Reader as there may be no new updates.
- Now go back into the customization menu and go back to the Shake menu. Turn off the Shake feature by choosing the Off button.

Task 2: Calling a friend

- Open the 4D application
- Open the contacts menu by pressing the lower panel. The menu should open automatically. Choose one of the available contacts to access their contact details. Press the phones standard back button to return to the application.

Task 3: Accessing and closing web pages

- Open the 4D application
- Slide the first panel across the page to access the web page. Navigate to one of the elements on the page, before sliding the panel back to its default position.
- Slide another web page from another panel of your choice, before sliding it back to its original position when you desire.

After completing all of the tasks above, please answer the questions below. You can re-do any of the above tasks again whilst answering the questions to refresh your memory.

Section 1- Questions based on 'Visibility of system status'

1. The headings and sub-titles are easy to understand and follow

   strongly agree agree uncertain disagree strongly disagree

   ☐ ☐ ☐ ☐ ☐

2. Menu instructions, prompts, and error messages appear in the same place(s) on each menu?

   strongly agree agree uncertain disagree strongly disagree

   ☐ ☐ ☐ ☐ ☐

3. High levels of concentration aren't necessary and remembering information is not required

   strongly agree agree uncertain disagree strongly disagree

   ☐ ☐ ☐ ☐ ☐

4. Whilst navigating the interface, it is always apparent to where you are?

   strongly agree agree uncertain disagree strongly disagree

   ☐ ☐ ☐ ☐ ☐

5. Whilst navigating the interface, it is always clear how to get back home?

   strongly agree agree uncertain disagree strongly disagree

   ☐ ☐ ☐ ☐ ☐

Section 2: Questions based on ‘Match Between System and the Real World’

6. Icons and buttons are similar to what I have seen in other systems.

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

7. The gestures used are similar to what I have used in other interfaces

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

8. A basic understanding of how to use the application is apparent without the use of instructions

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

9. On data entry screens, tasks described are in terminology familiar to you

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

10. Shapes for icons that have been used are obvious as to what they do

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

Section 3: Questions based on ‘Consistency and Standards’

11. Heavy use of all uppercase letters on a screen been avoided

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

12. Icons are labelled

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

13. Menu choice lists are presented vertically

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

Section 4: Questions based on 'Specific features'

14. The panels are a better way of navigating than other methods you have used

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

15. The contacts menu is easily accessible

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

16. The spinning logo gives you adequate feedback when updating

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

17. The shake feature is a suitable gesture for its purpose

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

18. The customization options are sufficient for your needs

strongly agree agree uncertain disagree strongly disagree

☐ ☐ ☐ ☐ ☐

Thank you for taking the questionnaire.

APPENDIX H- Full review of Handheld Mobile Device Interface Design (Gong, n.d; Tarasewich, n.d)

Introduction

This article by Jun Gong and Peter Tarasewich specifically addresses the problem of there being no dedicated heuristic guidelines for mobile design. It starts out by using Shneiderman's '8 Golden Rules of Interface Design' to propose a new set aimed at mobile design. The paper discusses the characteristics and limitations of current mobile device interfaces, especially compared to desktop environment (Gong, n.d.)

In this review I will be evaluating how they have used traditional heuristic principles and adapted these to suit a mobile interface. I will be analysing links between the previous literature reviewed such as Nielsen's 10 Heuristics and Ben Shneiderman's '8 Golden Rules of Interface Design'.

Enable frequent use of shortcuts

The first mobile design principle Gong (n.d) proposes is to 'enable frequent use of shortcuts'. This first originated in Shneiderman's heuristics but was aimed more at expert uses. Due to time being a critical factor in mobile usage Gong (n.d) explains that "Reducing the number of operations needed to perform regular (i.e., repetitive) tasks is a key factor in the ease of use of mobile devices".

Offer informative feedback

This principle found in Shneiderman's heuristics has had no adaption to the mobile interface guidelines. It is still recommended that messages should be written in plain English and not code.

Design Dialogs to Yield Closure

Again this principle has not been changed. "Users should be given the satisfaction of accomplishment and completion, no matter whether they are using desktop computers or mobile devices" Gong (n.d) explains.

Support Internal Locus of Control

This rule has very little adaption. The guideline appeals to both desktop design and mobile design. The extended advice for mobile design given by Gong (n.d) is to avoid using methods specific for mobile platforms where possible.

Strive for consistency

This principle has been modified quite significantly for mobile interface design. Gong (n.d) explains that "Consistency takes on an additional dimension with mobile applications… Users of mobile devices may need to switch between their desktop machines and different mobile devices frequently".

Reversal of Actions

This principle has also been modified. Gong (n.d) explains that reversal of actions is more difficult on mobile phones due to limited memory and resources, therefore it is harder for them to store past states.

Error Prevention and Simple Error Handling

Preventing and handling errors is similar to that of desktop design. Gong proposes that the need becomes more critical in mobile usage due to the rapid pace of events.

Reduce short-term memory load

The last adapted principle of Shneiderman's '8 Golden Rules of Interface Design' describes that users may have more distractions when using a mobile device, possibly hopping between applications regularly. There therefore means that the short term memory load becomes even less.

Extra guidelines

Gong (n.d) also proposes some extra guidelines for mobile interface design. Below I have summarized some of them.

- Allow for single- or no-handed operation
- Design for small devices
- Provide sound and tactile output options
- Allow applications to be stopped, started, and resumed with little or no effort
- Provide users the ability to change settings to their needs or liking

Sneiderman's original principle	**Adapted principle**
Enable frequent use of shortcuts	Reducing operations is a key factor in mobile design
Offer informative feedback	No adaption
Design Dialogs to Yield Closure	No adaption
Support Internal Locus of Control	avoid using methods specific for mobile platforms where possible

Strive for consistency	Users of mobile devices may need to switch between their desktop machines and different mobile devices frequently
Reversal of Actions	More difficult on mobile phones
Error Prevention and Simple Error Handling	more critical in mobile usage due to the rapid pace of events
Reduce short-term memory load	Users may have more distractions and may regularly switch between applications.

Fig.2.2: Table showing Sneiderman's original principle and Adapted principle