Undergraduate Honours Project

Interim Report

The Importance of Being Android

By
Daniel Muir
s0930256

Section 1: Introduction

Section 1.1: Original description of the Project

The project was to develop an Android application, primarily to help actors and actresses to learn lines for a play. For the project, the play; "The Importance of Being Earnest"[1] by Oscar Wilde was loaded into the app and, in addition to simply allowing someone to read the play, users may select a part in the play to learn. In this mode, the text of the play is shown up to their chosen character's next line and the user must click to advance the text. This enforced pause before their line allows the user to try to recall the line, before advancing to see if their recollection was correct.

In addition to making it easy to navigate through the text of the play, the app included the option of identifying cues in the previous line which prompt the user to help remind them what their next line is. It was required that the user could add performance notes to the text.

Section 1.2: Reasons for the Project

There were many reasons why developing such a project was something worth doing. First of all, it is easier for the user to learn the lines for their selected play at any time. Although this is already possible with a script on paper, having it stored on a device that is carried around by the user almost permanently, improves the efficiency of learning. It is also made easier for the user to quickly jump to different sections of the play. With the script on paper, the actor/actress must always search through the whole text when looking for their lines. With the app, the user is able to choose which parts of the script they would like to filter.

The most important reason is the idea of creating a "Digital Partner". Typically an actor or actress will first learn the lines of their play. Once they are confident in memorising the play, they will opt to run the lines through with another person so they can act off their performance. Since the app allows the user to select a character in the play, we re-create this experience without the need of another person. This greatly improves the learning experience.

Section 1.3: Finished State of Project

At the time of writing this report, the implementation of the project has been completed. In the coming days, the project will be extensively tested using the Android testing framework Robotium[2]. As well as this, the project will also be tested on a real device so it can be ensured the application is fit for purpose.

Once the state of the application is considered satisfactory, it will be passed on to some sample users who will complete a range of tasks and provide feedback on their

1 *The Importance of Being Earnest*, http://www.gutenberg.org/files/844/844-h/844-h.htm
2 *Robotium Toolkit*, http://code.google.com/p/robotium/

experience. The desired sample users for evaluation will be the members of the Edinburgh University Theatre Company.[3]

Section 2: Research and Design

Section 2.1: Research

Before beginning to think about design and implementation of the project, it was decided that some time should be spent on some background reading. This time was spent to find out how actors and actresses prepare for a role in a play, and see if the original specification of the project could be expanded upon.

Recording was a technique that was described as being useful for learning lines.[4] This was something that would work well with the application as the devices it would be used on has a recording feature built in. An extension on this idea was to allow the user to store the audio of their play onto their device so they could hear the other characters of the play while rehearsing.

Other ideas that came through research was to allow the user to highlight and breakup text.[5] Highlighting text is a popular technique to memorise lines as it allows the reader to focus on the words they have selected. This is particularly useful on an Android device as once the text has been highlighted, it can be removed. Breaking up the text allows the user to learn different sections of their lines separately which allows for an easier experience. This is particularly useful for long monologues.

A few other ideas were agreed upon after discussion with the project supervisor. It was agreed that the script of the play should be stored on an SQLite Database. The main focus of the app was to allow the user to quickly retrieve different parts of the play. Since databases allow for queries to quickly retrieve the desired data, this seemed like an effective data structure to use.

When an actor or actress is rehearsing for their play and they forget their line, someone will reveal some words and the rehearser can continue without breaking rhythm. This is a method that would also be included so again, rehearsing with the app could closely mirror rehearsing with others around you.

Although scripts won't change all that much, it was decided that the user would need some ability to edit scripts to accommodate for changes. As such, striking through text and removing stage directions was something that would also be introduced.

Finally the user would be able to display only their own lines for their selected character for ease of learning, and statistics were to be recorded to provide some feedback to the user.

3 *Edinburgh University Theatre Company*, http://www.eusa.ed.ac.uk/societies/society/bedlamtheatre/

4 *Acting: How to Learn Lines, Gedaly Guberek*, http://www.helium.com/items/822355-acting-how-to-learn-lines

5 *How to memorise your Lines: 10 Steps,* http://www.wikihow.com/Memorize-Your-Lines

Section 2.2: Tools required for development

Once the planning was complete of what would need to be done, a list was made noting what hardware and software tools would be required to successfully develop the project.

Software Tools:

- Eclipse IDE to develop the software
- Android development plug-in for Eclipse
- Image editing software for creating screen layouts
- Word Processing software for writing documents
- Version Control for backing up software (GitHub)
- Robotium framework for testing

Hardware Tools:

- Computer compatible with all of the Software Tools listed above
- A real Android device for testing. Multiple preferred

Section 2.3: Design

The last task that needed to be completed before implementation was to construct some designs to show how it was expected the application should look and perform.

Section 2.3.1: Home Screen

The Home Screen (Figure 1) would be where the user would start when they load the application. From here they would be able to go to any of the other screens in the app. To do this, the user can scroll through the text on screen which can be navigated through by pressing the up and down arrows.

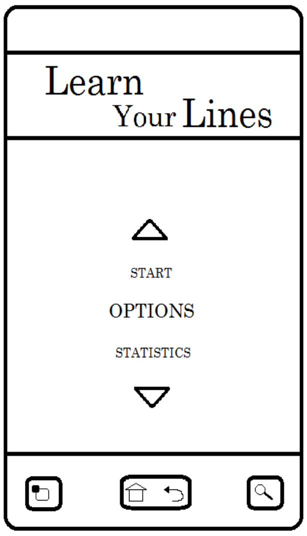


Figure 1: Initial design of the Home Screen

Section 2.3.2: Options Screen

The Options Screen (Figure 2) would be where the user would be next after pressing “Start” on the Home Screen. Here the user can select which character they wish to rehearse as, the act to begin from and also the page to begin from. One page represents roughly one A4 size of a sheet of paper. The user could also toggle if they wanted “Cue Words”, “Audio”, “Own Lines” or “Stage Directions” on or off. When the user would press “Continue”, they would proceed to rehearse their lines, based on their configurations.

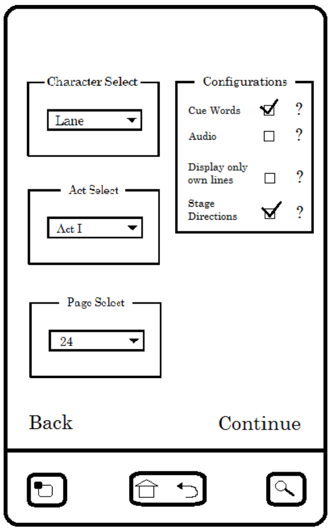
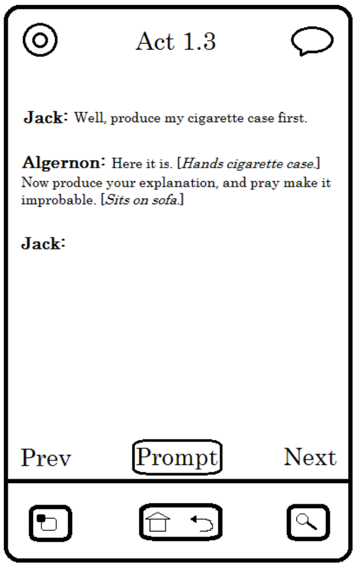


Figure 2: Initial design of the Options Screen

Section 2.3.3: Main Screen

The Main Screen (Figure 3) is where the user would spend most of their time. To reveal the selected character's current line, the user can press “Next”, and more lines will be revealed until the selected character's next line. Pressing and holding on “Next” will jump to the next page. Similarly, “Prev” will work in the same way, but backwards. “Prompt” will reveal the next hidden word from the current hidden line. At the top left of the screen is the “Recording” button. Pressing this will begin recording the user. At the top right is the “Performance Notes” button. Pressing this the user can create a new note.



Jack: Well, produce my cigarette case first.

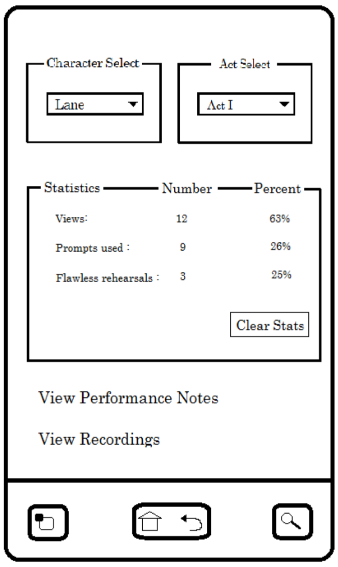
Algernon: Here it is. [*Hands cigarette case.*] Now produce your explanation, and pray make it improbable. [*Sits on sofa.*]

Jack:

Figure 3: Initial design of the Main Screen

Section 2.3.4: Statistics Screen

The user may view their stats any time on the Stats Screen (Figure 4). Similarly to the Options Screen, the user can select a character, an act or a page and view recorded stats based on their selection. The stats provided are the number of views, the prompts used, and “Flawless Rehearsals”. Flawless Rehearsals are the number of times the user managed to get through the current selection without requiring a prompt. The user can also view notes created and recordings made as well.



Statistics	Number	Percent
Views:	12	63%
Prompts used :	9	26%
Flawless rehearsals :	3	25%

Figure 4: Initial design of the Statistics Screen

Section 3: Implementation

Section 3.1: First Project Meeting

In preparation of the the first project meeting, the tasks set were to create a basic implementation of the four main screens that had been designed. First was the Home Screen. The only functionality that this screen would have at this point was to allow the user to navigate to the other screens in the app.

Most of the work came from developing the Options screen. As depicted in the original design, three Spinners were used to filter different parts of the script while four Checkboxes could toggle the different configurations. Since at this point the script of The Importance of Being Earnest wasn't being loaded into the app, the contents of each Spinner was just hardcoded. Also an “onItemSelectedListener” was added for the act Spinner so the relevant pages would be shown. Finally, a brief help guide was added for each of the Checkboxes, explaining what would change if they were enabled.

The Stats screen worked much in the same way as the Options screen with the three Spinners for filtering parts of the script. Some stats were hardcoded just to illustrate the look of the app. Finally a clear button was added which reset the stats.

The final screen to be implemented was the Main screen. Again since at this stage the script was being loaded, some hardcoded text was inserted to show how revealing a character's next line would work.

Feedback from the first meeting was overall positive. However, some were concerned that many features were being implemented without any evidence that they would be useful to those using the application.

Section 3.2: Second Project Meeting

Section 3.2.1: Crowd sourcing

The next task before continuing to do any more implementation was to think how to show that the features being implemented were going to be useful. To do this, it was required to get the input from those that worked in acting.

To gather the potential user's opinions, a survey was created using Google Docs. The user was asked to rate each of the following features:

- Filtering
- Prompt (reveal a word from hidden line)
- Cue Words (highlight words to hint to the user what their hidden line is)
- Recording
- Audio
- Performance Notes
- Highlighting
- Strikeout
- Breakup Text
- Statistics

Each of the above features could be rated as; *very useful, useful, don't know, unuseful or very unuseful*. Finally respondents were asked if they use any techniques that were not covered by the features they were asked to rate.

Section 3.2.2: Crowd Sourcing Results

Five responses were made to the survey and the results can be seen in the graph below (Figure 5). Overall the response was good to all of the features that were to be rated. Some of the features were a little debated, like the Statistics feature for example. However, all of the features received an average rating of at least *useful*. This justified that all the listed features should be developed.

Also when respondents were asked to provide their own techniques for rehearsing, three replied with the following insightful comments:

- “I think if the app could link to an osx native app that allows you to print out your scripts with the notes and additions made in the app that would be useful as well. My main problem is having multiple scripts with multiple versions of notes. It would be great to be able to print out a fresh one as rehearsals progress.”

- “random lines- the app gives random lines / the final words of lines of the other characters, not in order that they come in the script. This avoids the actor to learn 'chunks' of the script, and provides a more in depth knowledge of their cue lines.”

- “I personally find it easier to learn lines to songs, or tunes. So maybe have some tunes or songs that you can choose from. Just an idea. The rest of it sounds good though.”

Sometime was taken to consider these ideas, and whether they were worth developing. The decision was made later in the development process.

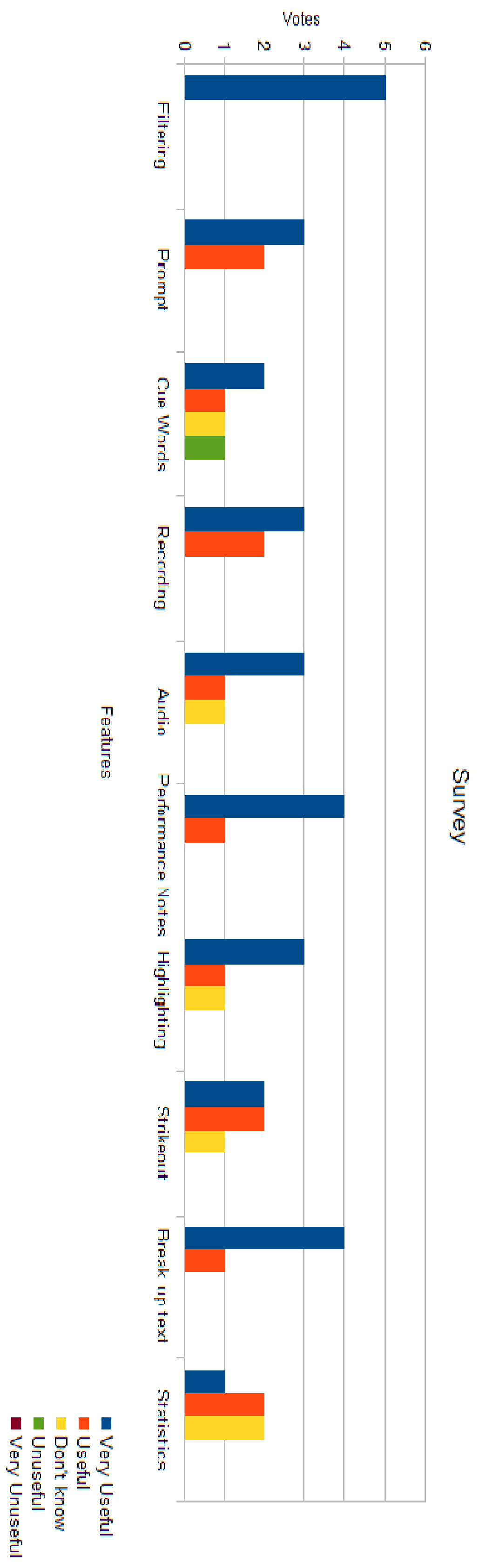


Figure 5: The results obtained from the survey

Section 3.2.3: Implementation for Second Meeting

Section 3.2.3.1: Database creation

In preparation for the second project meeting, much more development for the project was completed. The first thing created was the database that stored the script. The play was loaded into a standard text file, which was then placed in the Assets folder in the Project Directory. The entirety of the text file was then read by the app and the data was stored into the relevant parts of the database.

In order to make sure the database storing the script is populated correctly, the following strict rules must be adhered by when creating the text file:

- One line in the text file represents one line in the script
- All stage directions are within square brackets
- The name of the character must be immediately followed by a period. This is so the character speaking the line and the line itself can be differentiated.

As each line of the text file was being read, a new row was created into the database. The following is a list of the data that was contained in each row:

- A count of the line number
- Character speaking the line
- The line itself
- The act the current line was in
- The page the current line was in. After every 23 lines (which represented roughly one A4 page) the page number was incremented.
- Whether a note had been created for this line (Initialised to "N")
- Whether an audio file was associated for this line (Initialised to "N")
- The number of views for the current line (Initialised to 0)
- The number of prompts required (Initialised to 0)
- The number of "Flawless Rehearsals" completed (Initialised to 0)

Now that the database was fully populated, the script could be filtered based on the user's configurations.

Section 3.2.3.2: Filtering Script on screen

Now that the database was populated, work could begin on displaying the script to the user. The first task was to populate the Spinners on the Options screen dynamically. This was to allow the user to load a script of their choosing. For each Spinner, the database was read to get the results of all the different characters, acts and pages available. Duplicates were then removed and the remaining items were sorted. The character Spinner was sorted based on the number of lines each character had. The Spinners on the Statistics screen were similarly populated in this way.

The next step was to now display the script to the user based on their selections made in the Options screen. When the user chooses which page to start from, a query is made on the database to return all the lines on the given page. A "SimpleCursorAdapter" then displays the character delivering the line and the line itself in a “ListView”.

It was at this point a new Spinner was added on the Options screen. This was to allow the user to choose between “NORMAL” and “REHEARSAL” mode:

- NORMAL mode shows the script without any hidden text and statistics are not recorded.
- REHEARSAL mode again shows the script but with the selected character's lines hidden until the user proceeds. When in this mode, statistics would be recorded.

This Spinner was added so the user could first choose to learn the play at their own pace, before selecting a character and testing their knowledge of the script.

This user could also now toggle whether they wanted to display their selected character's own lines only. If this was toggled, the database would be queried to return the selected page and also the selected character's lines. It should be noted that the character selection Spinner was now always disabled unless “REHEARSAL” mode was selected or “display own lines only” was toggled on. Own lines could only be toggled if “NORMAL” mode was selected.

Finally the filtering of stage directions and the prompt feature was implemented. To remove stage directions, all text that appeared in between square brackets were removed before the script was displayed.

Section 3.3: Finishing Implementation

Section 3.3.1: Performance Notes

Now that the application could function on a basic level, the rest of semester was used to implement the remaining features. The first thing that was created was the Performance Notes feature. The user could long press on a line to bring up a Context Menu, which would allow them to create a new note. A dialog was shown which provided a text box to the user in which they could enter their note. The note was then saved to a newly created database. A new screen was implemented where the user could view all the notes they had written in a ListView. Finally, an icon was shown next to whatever lines had a note associated with it. Clicking the icon would display the list of notes associated with the selected line.

Section 3.3.2: Recording

The next major feature that was developed was the recording feature. Since I had no experience with recording on Android devices, some research was required.[6] Similar to how the performance notes was implemented, the user could press a button which would display a dialog in which the user could record themselves. Before saving, the user could preview the recording. Again, similar to performance notes, the list of recordings was displayed on a separate screen in a ListView.

6 *Android – Recording Audio*, http://www.youtube.com/watch?v=XANjoeEeQ1Y

Section 3.3.3: Audio

The next problem was how to develop the audio functionality. Originally the plan was to allow the user to download an audio version of their script and listen to that. However this wouldn't really be possible for most scripts, as audio may not exist, and could lead to some legal problems in obtaining the audio. To get past this, it was decided that it was up to the user to make the recordings, then they could apply the audio files to lines in the script. Once the user applied a recording to a line, they could select the line a playback the audio file.

Section 3.3.4: Statistics

The next task was to record statistics as the user navigated through the script, and display them correctly to the user when they chose to view them. The first decision to be made was when to increment the counters for each of the statistics' categories. After that, different queries were implemented that would obtain the correct data depending on the user's selections.

Section 3.3.5: Cue Words

The Cue Words feature was also developed. This compared two lines that followed one another, checked for duplicate words, and coloured the duplicates that appeared in the first line red. Duplicate words were also checked to make sure they were at least five characters long. This was to filter out words like “and” and “the”.

Section 3.3.6: Highlight and Strikeout

Unfortunately the Highlight and Strikeout features could not be implemented. The problem occurred when trying to allow the user to select the text they wish to either highlight or strike-through. There were two solutions to this:

- Create a customized EditText widget that would appear like a standard TextView, but would keep the text selection property.
- Upgrade the Minimum SDK version of the application to 3.0.

However neither solution proved satisfactory. Much time was spent creating a customized EditText but the result was very buggy and awkward to use. It was not possible to keep the selecting text functionality of a standard EditText but disable the context menu that automatically appeared. Also, there was no point upgrading the application to 3.0 as this would not allow many potential users with lower versions to use the app. Therefore the decision was made to abandon these functions.

Section 3.3.7: Settings

Finally, development for the first semester was complete with the creation of a Settings screen. On this screen, the user could select which script they would like to view, the number of words to reveal when they press the prompt button and also if they would like to enable auto-playback. The user configurations would be saved to a hidden settings file that would be stored on the phones SD card.

The list of scripts is displayed in a Spinner that is populated by the script files found on the phone's SD card. If the user decides to load a new script, the existing database would be deleted and a new one would be populated. Since populating a new database took sometime, a new Thread was launched which would display a Progress Dialog, instructing the user to wait.

The user could select to reveal either one, three, five or a whole sentence when pressing the prompt button. There was a problem when the user chose to reveal a whole sentence as it was difficult to identify what exactly a sentence was. Obviously a period indicates the end of the sentence but there would be some confusion in other instances such as when an ellipses were used. To get around this, a “BreakIterator” was used which successfully splits up a string into sentences.

Last was the auto-playback configuration. If this was toggled on, all lines on a given page with audio files associated with it would automatically play in sequence.

Another problem occurred here however. If there was more than one audio file in a given page, then all the recordings would playback instantly, and not wait until the previous one had finished. To fix this, a listener was added to ensure that the current audio file was finished, before playing back the next one.

Section 4: Future Work/Project Schedule

Task	Description	Deadline
1	**Finish Implementation.**	**End of Semester 1**
1.1	Design and implement Database.	November
1.1.1	Decide on what fields should be stored into database.	October Week 3
1.1.2	And filter feature to match user's character and page choice	November
1.2	Add Performance Notes functionality	November Week 2
1.3	Add Cue Words functionality	December
1.4	Add Prompt functionality	December
1.5	Add additional features	End of Semester 2
1.5.1	Store No. of prompts by user	December Week 1
1.5.2	Add Recording option	December Week 2
1.5.3	Add Audio option	December Week 3
2	**Test Project**	**February**
2.1	Use Android Testing Frameworks e.g. Monkey to test functionality	January Week 3
2.2	Evaluate project on different Android phones to test design	February
3	**Evaluate Project**	**February Week 3**
3.1	Get a range of users to use App	February Week 3
3.1.1	Provide users with tasks to accomplish	February Week 3
3.1.2	Provide users a Questionnaire to note their experience	February Week 3
4	**Write Dissertation**	**April**