iOS Report

Alex Roan

May 12, 2015

Contents

1 Design

For this project I've used a tabbed interface to control between adding and searching for word/phrase pairs, and the exercise for the user.

The first tab labelled "Translations" shows a list of all of the current stores translation pairs stored in the core data. The second tab labelled Revision displays all of the tags saved in the core data. It is by these tags in which the user can save new pairs, view pairs and revise from the already stored translation pairs.

In the core data, I've used the structure as described in the project specification, whereby the app is aimed at matching English and Welsh words and phrases.

2 Adding Pair Definitions

This was the first function that I implemented since it forms the core functionality of what all of the other requirements are based. Getting used to core data adding, retrieving and displaying data took some time but once this had been mastered, I found moving on and building on this was fairly easy.

The controller which controls the list of pair definitions is named PairsViewController. Upon loading it queries the core data for all of the pairs in the WordPhrasePair table and reloads the data in the prototype cells.

A push segue links these cells and view controller named NewPairViewController. This controller displays the details of a pair in several text boxes. When the add button is pressed on the PairsViewController, this NewPairViewController is displayed, with empty fields, so that a new pair can be added. Similarly, when an already saved pair from the list is clicked, the NewPairViewController is shows with the fields that have already been saved displayed. These fields are editable and can be re-saved if changes are made. All fields are not required to be filled in.

The user is also able to delete pairs from the table using the edit button in the top left. This action deletes the cell from the table, and also the record from the WordPhrasePair in the core data.

3 Tags

This was the second main function that I needed to implement. Once I was able to save data and retrieve data to the WordPhrasePair table in core data

I moved on to including the relationship between each WordPhrasePair and Tag.

Tags were slightly awkward to implement. When I first included them, every time a new pair was saved which included a tag or tags which were already present in the Tag table, it would add another instance. One way I could've worked around that was if I created a selector for tags which were already stored in the Tag table. Instead, due to time restrictions, I implemented a working check with capitalisation and trimming to tags which are split by commas in the input text box. This way, if the tag is already in the database, the new/edited pair will include the existing tag which will be capitalised. This is no less functional than if I used a selector for existing tags, ad in fact saves the user from navigating to a different tab to add or edit tags which do not exist. Instead, they simple type them in with a comma to separate them.

I then implemented the many to many relationship between the two, adding the WordPhrasePair object to the NSSet in Tag, and vice versa. This was probably the most difficult of the tag process, as it took some time to find out how exactly to use mutableSetValueForKey. Once this was completed, I moved on to searching

4 Searching

This was the third function which I aimed to implement. At this stage the app is able to add, retrieve and edit WordPhrasePair objects from core data. In doing this, the app recognises previously added Tag objects if a new pair is added with existing tags, or if an old pair is edited with and existing tag being added. It can also add new Tag objects to core data if they are not recognised, and obviously link the Tag object to the WordPhrasePair object in a many to many relationship

The search function is performed in the first tab in the PairsViewController. This was particularly difficult to implement because it makes use of two separate controllers, depending on which one is active at any given time. If the user is searching for a particular string of characters, then the SearchDisplayController is the active controller, and therefore, the value of the clicked row needs to be taken from the cells that remain in order to reach the detail page. If the search is not active then the normal UITableViewController is used to retrieve the row number to segue to the details page.

The search function checks whether the search string matches at all with both the English field and the Welsh field in the array of word/phrase pairs.

5 Exercise

In the Revision tab, all tags which are stored in the Tag table in core data are displayed in the list controlled by TagListController. Each cell is a detailed cell, and display not only the name of the tag, but also the number of associated word/phrase pairs along with it so the user knows how many they can revise in each tag category.

When the user clicks on one of these tags, the associated English words are displayed. Each one of these words are the revisable words in the revision section. If a user clicks one of these tags, they're taken to the RevisorController which displays the English version of the word at the top, and provides a text box for the user to input the Welsh version. If the user input matches the Welsh version in core data for that word, the user is prompted with correct, otherwise, incorrect.

This feature is a very simple implementation of it because of time restrictions and misreading the deadline dates. Had I more time I would have provided this feature for both Welsh to English, and the current English to Welsh revision.

6 Testing

Adding new word/phrase pairs works with every field. Adding tags to the tag field being separated by commas works and does check the database for previously compatible tags. Searching through the existing pairs works by attempting to match string using both the English and Welsh fields. Revising words works by comparing user input to the Welsh version of the word selected by the user. One thing which I have not implemented due to time restrictions was the ability to edit tags associated with a WordPhrasePair. Although the user is able to edit the fields within WordPhrasePair (English, Welsh, Note and Type), editing the tags associated with it will have no effect and the tags that were assigned at the point of creating the pair will stay the same.

I performed this testing using some data which I manually entered into the app. I then tested each of the functions one by one. Screenshots of the app working are available in the submission folder. Follow the README.txt for instructions.

7 Conclusion

All functions that were set out in the requirements specification have been completed. However, time restrictions caused by a misreading of deadline date meant that I completed the entire project in two days. This meant that unfortunately, the revision function has been implemented to minimum standards, but nevertheless completed.

Overall, I feel that this project should achieve something in the range of 65-75%.