WordMelodies: Supporting Children with Visual Impairment in Learning Literacy

Sergio Mascetti¹, Giovanni Leontini², Cristian Bernareggi³, Dragan Ahmetovic³

¹,²Università degli Studi di Milano
Diapartmento di Informatica
¹{firstname.lastname}@unimi.it
²{firstname.lastname}@studenti.unimi.it
³Università degli Studi di Torino
Diapartmento di Matematica
{firstname.lastname}@unito.it

ABSTRACT
We present WordMelodies, an inclusive, cross platform, mobile app that supports children with visual impairments in the acquisition of basic literacy skills through 8 different exercises. WordMelodies has been designed and evaluated by three domain experts in assistive technologies and education for children with visual impairments. After three design and evaluation iterations the app is fully accessible, except for one limitation of the cross platform development toolkit used.

Author Keywords
Visual Impairment, Education, Accessibility, Literacy;

CCS Concepts
•Human-centered computing → User centered design;
•Social and professional topics → Assistive technologies;

INTRODUCTION
Primary school educational tools are often not accessible for children with visual impairment or blindness (VIB) [3]. Indeed, textbooks are rich in graphical content, with the aim of engaging the students. However, this impacts their accessibility, even when they are available in a digital format. Similarly, educational apps often have inaccessible interactive graphical content, due to unsuitable design or sloppy implementation [7]. As a result, primary school education, including literacy skills is challenging for children with VIB [8].

To address these issues we present WordMelodies, an inclusive application to support children in the acquisition of basic literacy skills. WordMelodies was developed as a cross-platform mobile application and it includes 8 types of exercises. App analysis, design and evaluation were guided by three domain experts with a participatory approach. Following three design and evaluation iterations WordMelodies was considered to meet the functional requirements and to be fully accessible except for one issue, caused by the cross platform development toolkit, which we discuss in the following.

ANALYSIS AND DESIGN
WordMelodies has been developed with a participatory approach, involving three domain experts: a teacher for children with VIB, a congenitally blind expert in assistive technologies (who is also a co-author of this paper) and a primary school teacher. During the analysis phase, we created a list of 59 suitable exercises to practice literacy skills in primary school. To define this list we collected data from domain experts, teaching standards [6] and analysis of the existing apps and websites. For each exercise, we provided an interaction example and collected subjective evaluation by the domain experts on the exercise usefulness. We then selected the 8 exercises with the highest average usefulness score to be developed in WordMelodies. Figure 1 shows examples of exercises, the complete set is available online¹.

Another important aspect emerged while interacting with the domain experts: children in the target age group also need to develop key tech interaction skills [1]. In particular one expert remarked the importance of becoming familiar with Drag&Drop, a common gesture in mobile device interaction which is difficult to perform when using a screen-reader².

Based on this analysis, we defined the following criteria that drove the design and development phases.

• Inclusiveness. The app should be usable and easy to learn for children with and without VIB.
• Entertaining. Besides allowing the user to practice literacy skills, the app should also be fun and entertaining.
• Independence. The app should be usable by all users without requiring support from other people.
• Consistency. Key interaction elements should always be placed in the same part of the screen, possibly close to the screen corners or borders, where they are easier to find.
• Beyond tap. The app should use and teach common interaction gestures to children (e.g., Drag&Drop). The gestures should be introduced and explained. Note that this is the opposite of the “simple gestures” principle defined in [5].
• Scalable. It should be possible to add new exercises and content with limited developing effort.

1https://ewserver.di.unimi.it/wm/
2With screen-readers, Drag&Drop is done by double tapping the object to drag, keeping the finger down after the second tap, then moving the finger and releasing when on the target. On iOS, it is also possible to select and move an app using the actions item on the rotor
was to achieve an accessible and easy-to-use drag&drop functionality. Indeed, no basic React Native component supports accessible drag&drop. Hence, it was necessary to implement a React Native component on both Android and iOS to provide audio information during drag&drop when screen reader is active. The aim was to mimic the drag&drop behaviour exposed natively by both Android and iOS (e.g., when moving an app icon on the launcher). Rotor-based interaction, which is available only on iOS, was not replicated.

In WordMelodies the user selects an exercise type from a list. Then, one exercise instance is shown, among the available ones for that type. For example, when the user selects “listen and complete” exercise type, one of the two exercise instances shown in Figures 1a and 1b is shown. If the user makes a mistake, WordMelodies gives a negative audio and visual feedback. Instead, upon entering the right answer, WordMelodies gives a positive feedback; it plays a short piece of music and shows the next exercise instance for the same type.

In order to enhance the scalability and to ease the creation of several exercise instances, we developed WordMelodies to have a clear separation between the code and the data representing exercises content. Indeed, the exercise instances are defined within a standalone static JSON file. Thus, adding a new exercise instance does not require to modify the app code. To further extend this principle, exercise types have been grouped into two exercise classes: those based on the use of drag&drop gesture and those based on multiple choices questions. The exercise types in the same class are actually run by the same code and only differ in terms of static content. For example, both “day of the week” and “listen and complete” exercises (see Figures 1c, 1a and 1b) belong to the same class (drag&drop), while “rhyming words” (Figure 1d) belongs to the multiple choices class. This further enhances scalability, making it easy to add new exercise types as long as they present the same interaction as the existing ones.

One of the main challenges during the development of the app was to achieve an accessible and easy-to-use drag&drop functionality. Indeed, no basic React Native component supports accessible drag&drop. Hence, it was necessary to implement a React Native component on both Android and iOS to provide audio information during drag&drop when screen reader is active. The aim was to mimic the drag&drop behaviour exposed natively by both Android and iOS (e.g., when moving an app icon on the launcher). Rotor-based interaction, which is available only on iOS, was not replicated.

CONCLUSIONS AND FUTURE WORK

Through participatory design and multiple evaluation iterations with three domain experts, we developed WordMelodies, a cross-platform mobile app that supports children with and without VIB in learning basic literacy skills. The app is designed to be entertaining, inclusive and usable without external assistance. WordMelodies is also aimed at improving children basic tech interaction skills and teach the use of common gestures, such as Drag&Drop which is difficult to perform using a screen reader. Currently 8 different exercise types are available, with multiple instances for each exercise, and new exercise instances and types can be defined without modifying the underlying app code. While the app is completely accessible, exploration using flick gestures is currently not supported by the cross-platform development toolkit.

As a future work we will involve children with and without VIB and their parents in the app design process, in order to better adapt to user needs and interests. We will also work on the development of additional native components for iOS and Android to make flick-based exploration possible and consistent across different platforms. We will add new exercises, a storyline as an edutainment container for the exercises, and entertaining audio-visual content to engage children. Finally, we will distribute the app to end users.

ACKNOWLEDGMENTS

We would like to thank Diane Brauner for her invaluable support during problem analysis and system design.
REFERENCES


