MathMelodies 2: a Mobile Assistive Application for People with Visual Impairments Developed with React Native

Niccolò Cantù[1], **Mattia Ducci**[1], **Dragan Ahmetovic**[3], **Cristian Bernareggi**[2], **Sergio Mascetti**[2]

[1,2]Università degli Studi di Milano
Dipartimento di Informatica
[1]{firstname.lastname}@studenti.unimi.it
[2]{firstname.lastname}@unimi.it

[3]Università degli Studi di Torino
Dipartimento di Matematica
{firstname.lastname}@unito.it

ABSTRACT

Cross-platform developing techniques have been attracting lot of attention in the last years, especially in the field of mobile application, because they enable the developers to code apps in a same programming language for different platforms (e.g. iOS and Android). One well-known framework for cross-platform development is *React Native* that presents some features to support accessibility to blind or visually impaired (BVI) people. However, to the best of our knowledge, the accessibility of applications developed with this framework has not been systematically investigated.

In this contribution we report our experience in the development of *MathMelodies 2*, an application that supports BVI children to study mathematics. The former version of *MathMelodies* was developed with native code for iPad only, while *MathMelodies 2* was developed with *React Native* to run on both iOS and Android smartphones and tablets.

CCS Concepts

• **Human-centered computing** → **User interface design**, **Accessibility systems and tools**; • **Social and professional topics** → **Assistive technologies**, **People with disabilities**

Author Keywords

Visual Impairment; Blind Education; Math Accessibility;

INTRODUCTION

MathMelodies is an app that supports grade 1 to 5 children to learn mathematics [2, 1]. It is accessible and entertaining to sighted and BVI children and presents a fantasy tale in which the child solves math exercises to advance the story (13 types of exercises at different difficulty levels). *MathMelodies* was developed in 2013 thanks to a crowdfunding campaign and it is currently available for free on iPad[1]. Since its release, the app was downloaded more than 50,000 times worldwide, and in particular from English speaking countries and Italy.

[1] https://itunes.apple.com/app/math-melodies/id713705958

ASSETS '18 October 22–24, 2018, Galway, Ireland

ACM ISBN 978-1-4503-5650-3/18/10.

DOI: https://doi.org/10.1145/3234695.3241006

One common feedback from stakeholders was that they wanted the application to be available on different platforms, either the iPhone or on Android (both smartphones and tablets). To satisfy this request it was necessary to redesign the application to be usable on smartphones screen and to port the application on Android devices. In this contribution we report our experience in developing *MathMelodies 2*, a prototype application that addresses the above issues and that was developed with *React Native*, hence limiting the developing effort.

After presenting the main principles guiding the app design, we report our experience with the development of *MathMelodies 2* using *React Native*. We conclude that, while basic accessibility features are supported by this framework, one advanced capability, which is required by *MathMelodies 2*, is not supported. Thus, it is necessary to write additional native accessibility components. Despite this, the large part of the application is cross-platform. Preliminary tests with BVI users suggest that *MathMelodies 2* is fully accessible. Therefore, we conclude that *React Native* is a valid choice for the development of assistive applications.

DESIGN FOR SMARTPHONE AND TABLET

MathMelodies 2 was re-disegned to run on devices with different screen size (smartphones and tablets) both in landscape and portrait. This required to redesign the interaction flow in the app as well as most of the graphical interfaces. The design considered the following principles, which we derived from our own experience with the former version of the app and from the feedback received in the last years from the users:

- *Simple gestures.* The app should rely on simple gestures only, which can be easily entered also when the mobile screen reader (e.g. VoceOver or TalkBack) is active.
- *No-scrolling.* All elements should be visible on the screen without requiring the user to scroll the page, which is inconvenient for users that rely on screen readers or "magnifier".
- *Reference points.* Important interaction elements should always be placed in the same part of the screen, possibly in the bottom right part of the screen, where they are easier to access [4, 3].
- *Homogeneity.* The app should present the same interface to all users regardless of their disability (if any), and system accessibility tools should adapt the interface to user needs.

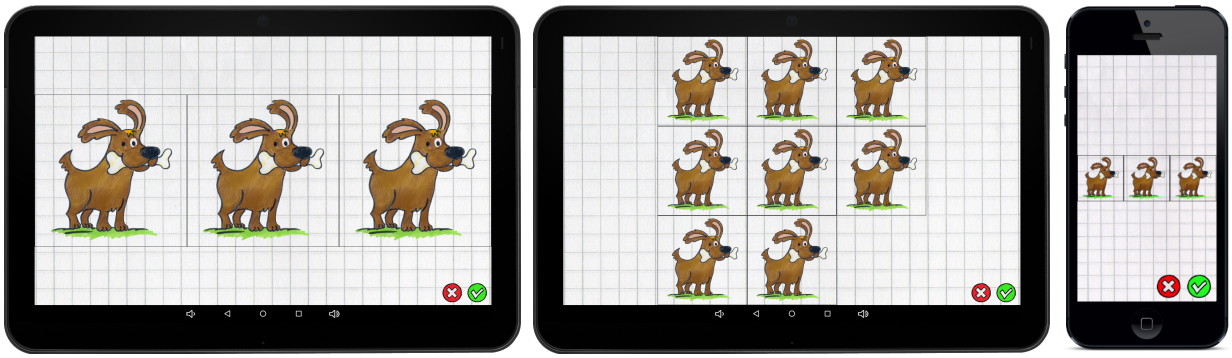

Figure 1: Counting exercise example. Left and center images: Android tablet. Right image: iPhone. Left and right images: easy version (grade 1) of the exercise with few animals to count. Center image: a harder version (grade 2 students).

- *Readability.* In order to enhance readability, only small text portions should be presented in each page, so that they can still fit in one screen if the "larger text" option is enabled at system level. Similarly, text background should be uniform and with neutral colors, so that it does not prevent readability both when normal and inverted colors are used.

While the above design principles were derived and extended from the former version of *MathMelodies*, there is one design principle that we had to adapt in the new design. In the first version of the app we used a *fixed-size* principle: all icons (e.g., those in Figure 1) have constant size throughout the application. This is convenient on tablet devices for blind users who explore the screen with a finger, and after some use can learn the icons' dimension. However, this approach also implies that, when an exercise presents a small number of icons, a portion of the screen is empty. Conversely, if there are many elements, a smartphone screen may not be able to contain them all, thus violating the *No-scrolling* principle.

While designing *MathMelodies 2*, we decided to adopt a *max-size* design principle, meaning that icons' size in each view is optimized to fill the entire screen. This approach makes it possible to show larger icons when few of them are present, but it also implies that their size changes in different exercises (see for example Figure 1). This choice was driven by the need of optimizing the use of the (possibly small) view.

DEVELOPING WITH *REACT NATIVE*

React Native maps its *components* to native graphical objects and makes it possible for the developer to specify basic accessibility features, like the alternative text for images. Consequently, most of the interfaces and functions of *MathMelodies 2* are accessible with system accessibility tools (like screen readers and magnifiers) and the developing effort to achieve this is analogous to the effort that would be required with native development.

However, some more advanced accessibility features are not supported by *React Native*. For example, in iOS and Android an event is fired when the screen reader focus changes. This event can be captured, when programming in native code, to perform an action associated to the focus change. Consider for example the counting exercise shown in Figure 1. When the screen reader is active, its focus changes when the finger moves from one icon to the other. Thus, it is possible, in native code, to play the dog's call each time a dog icon is touched. This way, a blind user can solve the exercise by exploring the screen with one finger and listening to the dog's call. Conversely, these events cannot be captured in *React Native*.

This is a typical problem in cross-platform development: to access system APIs that are available in native code but not through the cross-platform framework. A common solution to this problem is to develop plug-ins in native code that can interact with the cross-platform application. This is indeed the approach that we used to address the above problem.

We developed a native component (hence both in Android and iOS) that represents an image. This component looks and behaves like the *Image* component in *React Native* but it also forwards accessibility events (including changes in the screen reader focus) to a callback function in shared (i.e., JavaScript) code[2].

A prototype of *MathMelodies 2* featuring a subset of the story and of the exercises was tested by 2 grown-up users (one blind and one with low vision). Both users did not encounter accessibility problems on Android or iOS.

CONCLUSIONS AND FUTURE WORK

In this contribution we report our experience in the development of an assistive application with a well known framework for cross-platform development. We report some design principles that we believe can be useful for other similar applications and we described the technical problems that emerged during development and how we solved them.

As a future work we intend to develop all parts of *MathMelodies 2* so that it can be publicly released. Then we will be able to remotely collect usage data, which will make it possible to obtain a data driven assessment of app usability and accessibility.

[2]The developed software is available at http://everywarelab.di.unimi.it/15-research-projects/187-supporting-education#react

REFERENCES

1. Dragan Ahmetovic, Valeria Alampi, Cristian Bernareggi, Andrea Gerino, and Sergio Mascetti. 2017. Math Melodies: Supporting Visually Impaired Primary School Students in Learning Math. In *Proceedings of the 14th Web for All Conference on The Future of Accessible Work*. ACM, 26.
2. Andrea Gerino, Nicolo Alabastro, Cristian Bernareggi, Dragan Ahmetovic, and Sergio Mascetti. 2014. Mathmelodies: inclusive design of a didactic game to practice mathematics. In *International Conference on Computers for Handicapped Persons*. Springer, 564–571.
3. N. Harrington, Y. Zhuang, Y. O. Yazır, J. Baldwin, Y. Coady, and S. Ganti. 2013. Beyond user interfaces in mobile accessibility: Not just skin deep. In *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*. 322–329.
4. Shaun K Kane, Jacob O Wobbrock, and Richard E Ladner. 2011. Usable gestures for blind people: understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 413–422.