Methods in Creating the iBraille Challenge Mobile App for Braille Users

Cheryl Kamei-Hannan, Ph.D.
California State University, Los Angeles
ckameih@calstatela.edu
Tessa McCarthy, Ph.D.
University of Pittsburgh
tessam@pitt.edu
Benjamin Pomeroy
Braille Institute of America
BPomeroy@brailleinstitute.org

Abstract

Authors of the iBraille Challenge mobile app share wisdom gained during software development of a mobile app. An iterative, organic process was used to design the software and ensure accessibility for braille users. The app’s flowchart, research data, and details of how problems were addressed. These lessons learned may be used to pave the way for future accessibility of apps.

Keywords

Visual impairment, braille, IOS, APP, mobile device, universal design, assistive technology, ipad, braille display
Introduction

In recent years, tablets have become exceedingly prevalent in the public education system (Murray, 2011, Falloon, 2013, McMahon, 2014). Some schools, such as Westlake High School, have documented their adoption of tablet devices for all students (Foote, 2012). E-edition textbooks are distributed on these devices and numerous apps are used to enhance instruction. As teachers and students become more and more proficient with their use, they are finding innovative ways to use apps to support instruction (McMahon, 2014). In this revolutionary time of expanding tablet use and app development, universal design and accessibility is paramount to the construction of mobile apps. Some issues in accessibility are easily identified and may be addressed at the onset of software development. However, guidelines for software development are minimal (Sapp, 2007), particularly in relationship to mobile app development. Despite efforts to create universally designed software from the onset, many issues with accessibility are unknown until the app is in use. Similarly, teachers, students, individuals with disabilities, technology specialists, parents, and users of a particular device may identify a variety of different needs and options that make an app user friendly. In some cases, the app may be fully accessible, but successful use of the software requires advanced technology skills (Sapp, 2007), and the development team must simplify the user interface. To address ease of use, the opinions of a variety of different users, especially teachers, who have varying experiences and technology skills during development is critical (Falloon, 2013). The purpose of this paper, is to share the research-based, iterative, and organic process of development that authors used to create the iBraille Challenge Mobile App. Authors discuss the challenges that were identified from research and the problem solving approach they took to address these issues.
The iBraille Challenge

The iBraille Challenge, a 5-year project funded by the Office of Special Education Programs (OSEP), is an app created to support reading and writing development in children who read braille. It is delivered on an iPad paired with a Bluetooth refreshable braille display (an electronic screen reading tool that provides a braille display of the i-device’s visual output). The iBraille Challenge is based on the existing National Braille Challenge, an academic contest of braille readers and writers in which students compete against their peers on fundamental braille literacy skills such as reading fluency, reading comprehension, spelling and proofreading, and writing speed and accuracy. The purpose of the iBraille Challenge app is to build on the paper-based contest and enhance literacy skills of students who read braille by providing immediate feedback, monitoring progress over time, and addressing several targeted skills with activities.

Methods in App development

During the initial creation of the iBraille Challenge, authors began by assembling a team of experts who created the basic blueprint for the app. The team included 4 experts in braille literacy, 2 research experts, 2 technology experts, an individual who was visually impaired, 2 individuals representing agencies that serve individuals who are blind or visually impaired, a parent of a child with a visual impairment, and a child who was visually impaired. This team of experts determined the areas the app should address and identified issues in accessibility with the IOS device. They then created a storyboard for each activity that was going to be developed for the app. The storyboard served as the initial design element and included several pages, each page depicting a screen shot of the content and actions that would take place on each page.

The team considered accessibility at all stages of design. Considerations in the initial blueprint included determining the flow of the screens, organizing the items on the screen,
Methods in Creating the iBraille Challenge Mobile App for Braille Users

adoption a high contrast color schematic, using colors as visual indicators, placing buttons and essential information strategically, labeling/tagging of links and buttons, and routing the cursor for ease of the use (see Figure 1).

Because the app was designed to be dependent on Voice-Over (VO; the IOS’s built-in screen reader), standard IOS tables were used to organize and display information. The use of tables allowed for easy navigation using VO by reducing the amount of information presented on a page. Tables also allowed for a simple, high contrast, uncluttered screen. Another design feature that facilitated ease of use was the automatic routing of VO to specific areas on the screen.
screen. For example, if the user was required to choose an option from a list, then VO would automatically route to the first option on the list, thus eliminating the need to navigate through the headings of the table. Similarly, instead of presenting the user with options of pages to navigate, an intuitive forced selection model was used to automatically determine the order in which pages would be presented. For example, after completing an activity, the user would press the “finish” button and the app would automatically route the student to the next item within the same activity. The advantage of this forced routing was that users would not have to navigate through a clutter of options from which to select an item. The disadvantage was that if users needed to go to the beginning of an activity after they were half way through with it, they would have to navigate through several pages to go back to the beginning of the activity.

As part of the development, the technology team tested several configurations of accessibility settings, and identified the best configuration to use with the various activities in the app (see Figure 2; IOS 8.3). One important setting was the selection of the braille code to be used either English Braille American Edition (EBAE) or Unified English Braille Code (UEB).
In addition to the physical layout of the app, use, and flow of content within the app, the development phase included a sophisticated and integrated data management system that included multiple media sources designed to support the back-end of the app (see Figure 3).
First, because the app included grade levels 1st through 12th grade, an enormous amount of content was written (e.g. the reading comprehension section had over 700 passages, and each activity had over 200 individual activities) by a team of more than a dozen writers. A website was developed for these writers to input the content. Content from this website fed into a data chasse and information was housed in several layers, one for each component of the app. For example, reading comprehension, reading fluency, writing fluency, writing proofreading, and each activity had its own webpage and fed into a separate section within the data chasse. During
use, the internet dependent app downloaded the appropriate grade level content for the student from the data chasse. This was necessary in order to maintain adequate speeds for the content to load on the IOS device. Also, on the back end of the app, a data collection chasse was created to collect performance data for each of the students using the app. These data included the total amount of time the student used the app, the amount of time the student spent in each component of the app, and student performance within the contest and activities (e.g. words per minute for the fluency, types of reading errors made, number of questions answered correctly versus incorrectly, etc.). The data were used to answer research questions, such as “does the amount of time spent using the app make a difference in student performance?” Future research will include analyzing general trends in performance and a growth curve analysis of students’ performance.

Once the app was developed, research was conducted to determine the apps usability. An item-by-item analysis was used to evaluate the content and technical flow of each activity and each item within each activity. The analysis included a 10 question survey for each item and focused on the appropriateness of the content (e.g. non-biased, age/grade level appropriate, length), technological design (e.g. placement of buttons, correctly labeled tags, navigation of pages, automatic routing of the cursor), and braille output (e.g. accuracy of braille, navigability using VO with the braille display). Participants (n = 23) included braille literacy experts (n = 2), technology specialists (n = 4), individuals who read braille as their primary media (n = 7; 2 of whom were also teachers), teachers (n = 7), an individual with low vision (n = 1), a parent (n = 1), students (n = 5, all of whom were braille readers), and representatives from organizations that serve individuals who are blind and visually impaired (n = 2). These participants completed the item-by-item analysis and provided general feedback about the app including the ease of use (e.g. flow of the app, organization of content). All participants underwent three hours of training.
that included basics of VO, pairing a braille display to the iPad and using the braille display to navigate the IOS device, and using the app. Teachers worked with their students by implementing the app in weekly instruction for four to six weeks. All participants were given ongoing technical support and additional training was provided to those who needed it.

**Findings and Design Revisions**

Results indicated 80% of the participants rated the app as highly useful and of high quality, and 98% of the participants felt that the app addressed braille literacy development. One outcome of the data indicated that proofreaders should be hired to identify editorial issues (e.g. consistent use of capitalization, punctuation, ability to derive one correct response, elimination of duplicate answers). Other outcomes included that the braille display was not accurately translating all of the content, and the programmers of the app had to manually override certain characters in order for the IOS device to translate them correctly. The second and third stages of research are ongoing. The second stage will focus on the efficacy of the app and the third stage will focus on mass dissemination.

Specific issues came about when testing the app’s accessibility. First, teachers needed to be able to navigate the app using a braille display. A switch was inserted on the homepage of the app that indicated that a teacher was using VO. This functionality was important because the audio settings used in the app needed to be altered when VO was being used. Also, because the IOS device cannot record and playback audio clips while using Voiceover, an alternative method of recording needed to be created for teachers to use this function in the app. Users also commented on the order in which VO read the labels for the buttons and switches, and a consistent schematic for announcing buttons, switches, and headers was created. Similarly, the instructions needed to be more explicit and they were edited to address the users concerns.
Another finding was that certain braille characters were not displaying accurately on the braille device. The quotation mark was one such character. When writers imported content from the word processing software, the use of a print font with “curly quotes” created an unusual symbol in braille. Thus, the app developer needed to manually program an override for quotation marks which transformed the curly quotes into straight quotes. Similarly the use of the Letter Indicator (LI), a symbol used in braille to indicate a single letter as opposed to a whole word alphabet contraction, was not accurately translated in all situations. For example, if a single letter was used followed by a period – “Go to room A.” – then the LI was not used, but if it were followed by a comma or a semi-colon it rendered fine in braille. In this situation we rewrote the content to never have the single letter followed by a period.

**Discussion**

Educational tablet software developers should consider the needs of individuals with visual impairment and blindness, including those who read braille. Having individuals who read braille test the software during the development stage is essential. Development of this mobile app presented unique issues dealing with the braille code that were discovered by proficient braille readers who were integral during the initial stages of development.

One issue had to do with the need to purposely create text with errors in it. This was required of an activity that tested a student’s ability to proofread text. The activity included sentences with errors in them and the student was asked to select the sentence that did not have an error in it. Creating the content for this activity required writers to purposely write sentences with errors. This posed a particular challenge with the IOS translation software which would automatically correct some of the braille errors. For example, if the test writer wanted to test the student’s ability to recognize that a word that was incorrectly contracted in braille, the IOS
device would contract it [e.g. if the student was asked to identify a word that was incorrectly contracted as bro(th)(er), the IOS device would automatically correct it to read bro(the)r]. Using the uncontracted braille setting and inputting content in ASCII was a method used to force braille errors. In this activity, the app visually displayed the braille font in simulated braille. The use of the braille font was novel to app development, and posed a few issues with the ASCII assignments of braille symbols. For example, dots 5, 6 and 4, 6 did not have an ASCII equivalent, and therefore would not show in simulated braille. Programmers had to assign a keyboard stroke that was not being used to simulate these anomalous characters. Lastly, also related to this issue was the need to write some activities so that contractions would not be used (See Figure 4).
Methods in Creating the iBraille Challenge Mobile App for Braille Users

Fig. 4. Braille Hunt
For example, one activity, entitled Braille Hunt, was designed to help students improve their tactile ability to discriminate between similar braille letters. In this activity, a string of characters was shown that included one anomalous character (e.g. find the letter g in the line hhhhhghhhhhghhghg). The student was asked to search for the anomalous character and enter a key stroke (braille a full braille cell using dots 1, 2, 3, 4, 5, and 6). Only through testing using the braille display was it revealed that the IOS device would automatically translate the “gh” in the middle of the string of characters and render the braille contraction for the letters “gh.” Because the activity evolved into searching for braille contractions and the instructions in the activity needed to be in contracted braille, the uncontracted portion of the activity had to be rewritten using characters that did not render contractions.

Conclusion

With only 5 years of experience in IOS app development, since the introduction of iPads, we are at the forefront of app development and accessibility. Moving forward, the field of visual impairment should form a network to identify issues in accessibility and solutions to these issues. Discussions should include topics such as how to use build apps that are compatible with the built-in accessibility features of the IOS device, such as VO and design features that enhance usability. Ultimately, stakeholders in the field of visual impairment and blindness should adopt guidelines for app developers to follow. Guidelines are in place for websites, and a website, http://www.bobby-approved.com has been developed to test accessibility of webpages (Cogan, 2015). Similar guidelines are needed for app development.

As issues accessibility, including IOS compatibility with braille, are discovered, app developers should develop a network to share information. Several apps are being developed to support learning of students who are visually impaired. The information learned as these apps
are developed could potentially assist all software developers who are creating instructional software. Such a network would also be instrumental in developing guidelines for app developers to follow. Additionally, consumer organizations and organizations representing individuals who are blind or visually impaired should work with manufacturers to improve functioning of VO and braille translation.
Works cited


