Designing for Tinkerability and Accessibility

Developing the OctoStudio mobile app to engage blind and visually impaired learners in creating with code

Katarina Bulovic
kbulovic@media.mit.edu
MIT Media Lab, Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

Zoë Bentley
zmb@media.mit.edu
MIT Media Lab, Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

Natalie Rusk
nrusk@media.mit.edu
MIT Media Lab, Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

ABSTRACT

Millions of children around the world learn to code by creating with Scratch and other block-based programming languages. However, these programming environments typically are not accessible for blind and visually impaired children to tinker, create, and learn alongside their sighted peers. This paper discusses the ongoing development of the OctoStudio coding app to support accessibility and tinkerability for blind and visually impaired learners. We discuss how we have applied core principles of tinkerability to create an accessible, mainstream app for use on mobile phones and tablets. We describe our iterative development process in collaboration with educators who specialize in the design and testing of accessible technologies for children. We conclude with suggestions for how the core principles of designing for tinkerability can be expanded to support accessibility and engagement of blind and visually impaired learners internationally.

CCS CONCEPTS

• **Human-centered computing → Accessibility**; • **Applied computing → Interactive learning environments**; • **Social and professional topics → Computing education**.

KEYWORDS

Block-based programming, Accessibility, Screen Reader, Tinkering, Blind and Visually Impaired Learners.

ACM Reference Format:
Katarina Bulovic, Zoë Bentley, and Natalie Rusk. 2024. Designing for Tinkerability and Accessibility: Developing the OctoStudio mobile app to engage blind and visually impaired learners
in creating with code. In *Interaction Design and Children (IDC '24), June 17–20, 2024, Delft, Netherlands.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3628516.3659411

1 INTRODUCTION

OctoStudio is a new free coding app designed to make it easy for children to create interactive projects on mobile devices [16]. OctoStudio is developed by the Lifelong Kindergarten group at MIT, building on the group's experience designing creative learning environments for children, including the Scratch programming language [15]. While Scratch and related creative coding environments engage millions of children around the world in tinkering and creating with code, a fundamental limitation is that most of these child-friendly creative coding environments lack support for blind and visually impaired learners.

From the beginning of the design process, a core motivation for creating OctoStudio was to make creating with code more accessible to learners with a variety of backgrounds and interests. OctoStudio was designed in collaboration with educators from Brazil, India, South Africa, Mexico, and other areas to work well for children in their communities [16]. The app is designed to work on mobile devices with limited storage and screen size and does not require WiFi or internet connection. Learners can create a wide variety of projects using built-in sensors and screen reader functionality on mobile devices without needing additional hardware.

OctoStudio launched in October 2023 with basic accessibility features and was announced on the Perkins School for the Blind website [8]. The OctoStudio app is freely available for Android and iOS phones and tablets. All text within the app meets the WCAG 2.0 standards for color contrast [18], and the app is navigable with a screen reader. However, accessibility in OctoStudio is still a work-in-progress, and we continue to iterate and make improvements to more fully support all learners.

In this paper, we describe how we developed OctoStudio to support both tinkerability and accessibility for blind and visually impaired learners. The paper is based on research by Bulovic (first author) in her master's thesis [4]. We also describe additional improvements informed by screen reader testing and illustrate ways to apply principles of designing for tinkerability to develop more accessible technologies.

2 RELATED WORK

2.1 Designing for Tinkerability

Learning through tinkering is an approach designed to support learners of diverse abilities and backgrounds in actively making projects through playful experimentation, exploring their ideas, and discovering new possibilities [6]. In their widely cited paper, "Designing for Tinkerability", Resnick and Rosenbaum [14] offer three core principles for designing construction kits for tinkerability. First, the user should be given *immediate feedback* as they make changes. Second, *fluid experimentation* should be encouraged; it should be easy to get started and to fluidly connect and disconnect materials. Finally, the creative tool should inspire *open exploration* by providing support for a wide variety of projects. These three principles are described and illustrated with examples of how Scratch has been designed to support playful creative learning, with a focus on providing visual feedback [14]. For blind and visually impaired (BVI) users, these three principles of tinkerability are just as important, but adaptation is required to achieve these principles with non-visual methods. The research described in this paper addresses those challenges so that BVI learners can use tools such as the built-in screen reader to create with OctoStudio in a tinkerable way.

2.2 Designing for Accessibility

A variety of tools have already been made to support developers in designing for accessibility. For example, current computers and smartphones come with a built-in screen reader. The function of a screen reader is to present visual content on the screen through audio descriptions [5]. There are also tools designed for developers to work with the screen reader, such as ARIA, a toolkit which supports adding accessibility features to web applications [9]. In addition to ensuring screen reader compatibility, there exist standards for designing applications for blind users, such as adding alternate text to elements that need additional description and assuring that the screen reader announces visual changes to page content [18].

Block-based programming environments, such as Scratch, are commonly used to teach coding, but most are not accessible to BVI learners due to their reliance on the mouse and their lack of compatibility with screen readers [1]. Various programming environments have been developed to address this issue of accessibility. Some text-based environments allow for coding with a screen reader [2], but are less tinkerable due to their strict syntax. Quorum Blocks [3] is a more tinkerable, block-based language developed with a focus on BVI students, though it uses Java syntax, which is not as beginner-friendly as a language like Scratch [13]. Recently, accessible versions of block-based coding libraries, such as "Accessible Blockly" [1, 17] have been developed, although these are not yet designed for children to engage in creating projects on mobile phones and tablets. Several accessible tangible programming environments and block-based mobile apps focus on robotics or require other specialized hardware (e.g., Blocks4All [12], CodeJumper [10]). There are also accessible mobile apps (e.g., CodeQuest [11]) that focus on solving challenges or puzzles, rather than creating projects.

OctoStudio is designed to create an open-ended and child-friendly programming environment that enables BVI learners to engage in creative expression in a widely available free app, building on the principles of tinkerability and making use of the accessibility features that are built into smartphones and tablets. Using built-in functionality is especially important for expanding access for children across communities around the world.

3 DESIGN TEAM AND PROCESS

The authors of this paper are part of a larger team developing OctoStudio within the Lifelong Kindergarten research group at the MIT Media Lab. Katarina Bulovic (first author) has led the research and development of accessibility features on the app, and is a member of the OctoStudio software engineering team. Zoë Bentley (second author) leads the development of accessible resources and testing at OctoStudio, and is visually impaired. Natalie Rusk (third author) is project director for OctoStudio.

The design of OctoStudio accessibility has been guided by educators with expertise in screen readers and other accessible technologies for blind and visually impaired learners. Our two main accessibility advisors are Diane Brauner and Linda Coccovizzo. Diane Brauner is an educational accessibility consultant who specializes in the design of accessible educational software for children. Linda Coccovizzo is an Adaptive Technology Instructor, specializing in screen reader testing and accessible technology for children. She has been blind since birth and uses a screen reader herself.

Early in the process of designing for accessibility in OctoStudio, Diane Brauner recommended using the built-in screen reader on iOS and Android devices. She also recommended making a single mainstream app for both sighted and BVI learners to support inclusion and peer learning. Linda Coccovizzo has conducted in-depth screen reader testing on both Android and iOS devices and provided feedback on multiple iterations of the app. As we implemented and improved screen reader compatibility based on testing and feedback, we also identified ways in which OctoStudio was geared towards sighted users, including focusing on visual cues, feedback, and tools. As described below, we have worked to fix many of these issues and are now ready to playtest with BVI learners and gather feedback to make OctoStudio more enjoyable, expressive, and tinkerable without seeing the screen.

4 SCREEN READER FUNCTIONALITY IN OCTOSTUDIO

We have designed OctoStudio to work with screen readers on both iOS and Android devices (called VoiceOver on iOS or Talkback on Android) that can be turned on in the device settings. VoiceOver and Talkback allow a user to navigate apps and interact with the page without needing to see the screen. As the user drags their finger around the screen, the screen reader will read aloud any text or interactive page elements that are at that location. Alternatively, the user can swipe left or right to jump to the previous or next element. In addition to reading out loud what is underneath their finger, the screen reader will also automatically change the current "focused element" to whichever element the user is hovering over. This allows the user to interact with that element. For example, if a user drags their finger over a button, the screen reader will read the text on the button and focus on it. Once the button is focused, the user can then double tap anywhere on the screen to press the button.

Our first step was to test OctoStudio with built-in screen reader functionality. Although some accessibility features worked automatically (such as screen readers recognizing native HTML elements like buttons and links), there were other features and page elements that screen readers did not recognize. For instance, there are places in OctoStudio where native HTML elements lack the complexity needed for a custom button or section of the page. In these cases, generic HTML `<div>` elements are used, and are given code and styling to reach the desired appearance and behavior. However, the screen reader by default did not know how to interact with these custom elements and would read unhelpful information or no information at all when trying to focus on them. To address this problem, we used ARIA properties to give both roles and labels to these custom HTML elements, allowing the screen reader to accurately describe them to users [9] (e.g., we improved accessibility of the play button by giving it the properties `role="button"` and `aria-label="play"`).

5 APPLYING PRINCIPLES OF TINKERABILITY FOR ACCESSIBILITY

An important first step to make an app accessible is usability with a screen reader. However, this alone is not enough to guarantee a meaningful and enjoyable experience for BVI learners. To provide an engaging creative learning experience with OctoStudio, we applied the three core principles recommended in "Designing for Tinkerability" [14]. In this section, we describe our approach to applying and adapting these three principles: (1) immediate feedback, (2) fluid experimentation, and (3) open exploration to support both tinkerability and accessibility for BVI learners.

5.1 Immediate Feedback

The first guiding principle presented in the "Designing for Tinkerability" paper is *immediate feedback*: people should be able to immediately "see" the process and results of their creation [14]. For example, in OctoStudio, sighted learners can see the process by watching code blocks highlight as they run and can watch sprites animate on the screen. To be accessible to BVI learners, feedback should allow people to experience the process and experience the result in ways that do not rely solely on visual information.

One way to accomplish this goal is through sonification. Sonification is "the use of non-speech audio to convey information or perceptualize data" [7]. In OctoStudio, we added sonification to associate each visual output block with a short auditory cue. For example, whenever the "jump" block runs and the character jumps on the screen, a "boing" sound is played. A "glow" block shows a glow around the sprite and makes an electric hum sound. Every block with visual feedback has an associated sound, which allows BVI learners to get immediate feedback at the same time as their sighted peers.

Designing sound effects for sonification was an iterative process, with rounds of feedback from our screen reader consultants as well as our internal team. The sounds needed to be informative, so that the user could understand what was happening visually, but also engaging and fun to listen to even when played repeatedly in a loop. Also, it was important to keep the sounds short so that they could run as the program ran in real time. An example of this iteration was our design process for the "tilt to move" block, which allows the user to tilt the phone to drive a character around the screen. We started by playing a continuous tone while this block was running which would change in frequency based on the y-position of the sprite. This gave the learners some information about the sprite's position, but it was not pleasant to listen to. We improved on this sound effect by switching from a continuous tone to a series of soft staccato notes that sound like footsteps of the sprite moving around, which now conveys information about the position and direction of the moving sprite in a playful and meaningful way.

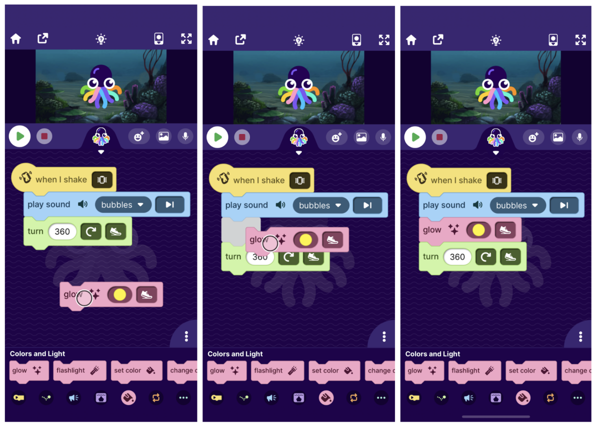

Figure 1: Using drag-and-drop to add a new coding block to a project. The learner receives feedback via screen reader announcements and sound effects as they pull out a new block (left), choose where to place it (center), and as they drop it into place (right).

5.2 Fluid Experimentation

The second guiding principle for designing for tinkerability is *fluid experimentation* [14]. Tinkerable coding environments should make it easy to assemble and take apart elements, and easy to get started creating projects.

5.2.1 Easy to Assemble and Take Apart. Like Scratch, OctoStudio makes it easy for sighted users to assemble coding blocks, using shapes to visually indicate which blocks will fit together and highlighting areas where a block can connect. In OctoStudio, we have additionally developed accessible sound-based drag-and-drop features. When a user turns on their screen reader and starts dragging a block into the coding area, the screen reader will announce valid drop areas. If the user is dragging a block of code and they hover it over another block it can connect to, the screen reader will announce the name of that block. This way, a screen reader user can easily identify blocks that can connect and fluidly manipulate them to build programs, without needing to see the block shapes (Figure 1). To further improve accessibility for drag-and-drop, we automatically snap the blocks into a vertical line when the screen reader is turned on. This allows a learner to find the block they want by simply dragging their finger down and listening as each block is announced by the screen reader, instead of having to search all over the screen.

5.2.2 Easy to Get Started. To help sighted beginners get started, OctoStudio includes a one-minute Getting Started video with many different examples for inspiration. Although this video has an audio description track that narrates the visuals for accessibility, our advisors said the video was overwhelming because it rapidly switches between many different examples. They suggested developing a new video tutorial specifically focused on the screen reader experience. They recommended making a video rather than an audio file because this will provide BVI learners with human narration and at the same time provide visuals for sighted educators who may support BVI learners. We have designed the screen reader tutorial as a walkthrough of creating a single project, which explains the layout of the app while limiting content to avoid information overload. We developed the script with our advisors and are currently producing the video, which we will then try out as part of playtesting and gathering feedback from BVI learners.

5.3 Open Exploration

Finally, the paper "Designing for Tinkerability" emphasizes the value of facilitating *open exploration* in creative learning experiences [14]. Learners should be provided with a variety of materials with which they can tinker and build their creations, and a variety of project genres to explore and create based on their own interests.

To ensure OctoStudio provides a variety of materials and enables creating projects in a variety of genres, we started by identifying two main areas to focus on to support meaningful engagement for BVI users. The first was to diversify the kinds of materials provided to learners in OctoStudio. The app provides libraries of media assets for characters, backgrounds, and sounds that can be added to projects. In addition to having a paint editor where users can draw images, we also have a sound editor where users can record and tinker with sounds. Our sound editor adds the ability to crop sounds and change sound properties with various filters, such as playing with the speed of their voice. These sound controls facilitate the same kinds of rich, complex interactions with audio assets that the paint editor provides for visual assets.

The second focus area was to provide coding blocks that are not solely for visual output. OctoStudio includes two blocks that play audio: the "play sound" block (which plays a recorded sound) and the "speak" block (which performs text to speech). Additionally, the "buzz" block causes the device to vibrate with haptic feedback. The app also includes sensing blocks that can be used to build non-visual projects. Users can choose to start their project running by shaking the phone, tapping anywhere on the screen, or bringing a magnet close to the device. The current tilt orientation of the phone can also be used to move characters around the screen. To further facilitate open exploration for BVI programmers, we organized these non-visual blocks to be among the first few available blocks in their respective categories.

6 FUTURE WORK

Accessibility for BVI learners in OctoStudio is still a work in progress. Some areas of in-progress work include allowing editing of block parameters for screen reader users, facilitating more ways to tinker with sound, multilingual translation of screen reader text, and developing screen reader learning resources. Another major area of planned work is to expand our iterative design process to include more educators and BVI learners. We are especially interested in feedback from BVI children about their experience tinkering with the OctoStudio app to create projects based on their ideas and interests.

The core principles of designing for tinkerability are helpful for designing creative technologies that actively engage children with diverse interests and abilities in learning through making [6], but are often described and applied in ways that focus on visual representation [14]. Our experience through the iterative development of OctoStudio illustrates how the core principles can be applied to design creative coding environments that are inclusive and accessible for blind and visually impaired learners. We suggest expanding the description and application of the core principles of designing for tinkerability to encourage the development of creative technologies that are both tinkerable and accessible. The principle of immediate feedback can extend beyond relying solely on visual feedback, so that learners can experience the process and experience the result—for example, combining screen readers with sound effects that reflect the playful process while providing meaningful feedback. Similarly, designers can apply the principle of fluid experimentation to enable learners to easily get started and to combine and take apart elements throughout their creative process by leveraging diverse features and functionality in existing technologies (e.g., voice output, sensors, and haptic feedback). Applying the principle of open exploration to support greater inclusion and accessibility suggests providing materials and tools that enable learners with diverse abilities and interests to make a wide variety of creations with diverse types of media (e.g, providing sound editing tools in addition to image editing tools).

REFERENCES

[1] Aboubakar Mountapmbeme, Obianuju Okafor, and Stephanie Ludi. 2022. Accessible Blockly: An accessible block-based programming library for people with visual impairments. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '22)*. Association for Computing Machinery, New York, NY, USA, Article 19, 15 pages. https://doi.org/10.1145/3517428.3544806

[2] Aboubakar Mountapmbeme, Obianuju Okafor, and Stephanie Ludi. 2022. Addressing accessibility barriers in programming for people with visual impairments: A literature review. *ACM Trans. Access. Comput* 15, 1, Article 7 (2022), 26 pages. https://doi.org/10.1145/3507469

[3] Andreas Stefik, Willliam Allee, Gabriel Contreras, Timothy Kluthe, Alex Hoffman, Brianna Blaser, and Richard Ladner. 2024. Accessible to Whom? Bringing Accessibility to Blocks. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 1286–1292. https://doi.org/10.1145/3626252.3630770

[4] Katarina Bulovic. 2022. *Designing for Tinkerability for Accessibility.* Master's thesis. Massachusetts Institute of Technology.

[5] Alastair Craig. 2019. *Designing and developing your Android apps for Blind Users (part 2).* Retrieved March 24, 2024 from https://medium.com/@AlastairCraig86/designing-and-developing-your-android-apps-for-blind-users-part-2-6caae0d4a687

[6] Daniela K. DiGiacomo and Kris D. Gutiérrez. 2016. Relational equity as a design tool within Making and Tinkering activities. *Mind, Culture, and Activity* 23, 2 (2016), 141–153. https://doi.org/10.1080/10749039.2015.1058398

[7] Perkins School for the Blind. 2021. *Sonification Summary Page.* Retrieved March 25, 2024 from https://www.perkinselearning.org/technology/blog/sonification-summary-page

[8] Perkins School for the Blind. 2023. *Announcing OctoStudio: A Block-Based Programming App.* Retrieved March 25, 2024 from https://www.perkins.org/resource/announcing-octostudio-a-block-based-programming-app/

[9] Mozilla Foundation. 2024. *ARIA.* Retrieved March 25, 2024 from https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

[10] American Printing House. [n. d.]. *Code Jumper.* Retrieved March 25, 2024 from https://codejumper.com/

[11] American Printing House. 2024. *CodeQuest (iPad App).* Retrieved March 25, 2024 from https://www.aph.org/product/code-quest-for-ipad-only

[12] Lauren R. Milne and Richard E. Ladner. 2018. Blocks4All: Overcoming accessibility barriers to blocks programming for children with visual impairments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article 69, 10 pages. https://doi.org/10.1145/3173574.3173643

[13] Michal Armoni, Orni Meerbaum-Salant, and Mordechai Ben-Ari. 2015. From Scratch to "Real" Programming. *ACM Trans. Comput. Educ.* 14, 4, Article 25 (2015), 15 pages. https://doi.org/10.1145/2677087

[14] Mitchel Resnick and Eric Rosenbaum. 2013. Designing for tinkerability. In *Design, make, play: Growing the next generation of STEM innovators*, Margaret Honey and David E. Kanter (Eds.). Routledge, New York, NY, 163–181.

[15] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. 2009. Scratch: programming for all. *Commun. ACM* 52, 11 (2009), 60–67.

[16] Natalie Rusk, Rupal Jain, and Caitlin K. Martin. 2023. Designing for belonging: Partnering with community-based educators to develop a new app for creative expression. In *Proceedings of the 17th International Conference of the Learning Sciences - ICLS 2023*. International Society of the Learning Sciences, 2025–2026.

[17] Stephanie Ludi and Mary Spencer. 2017. Design considerations to increase block-based language accessibility for blind programmers via Blockly. *J. Vis. Lang. Sentient Syst.* 3, 1 (2017). https://doi.org/10.18293/VLSS2017-013

[18] W3C. 2008. *Web Content Accessibility Guidelines (WCAG) 2.0*. Retrieved March 25, 2024 from https://www.w3.org/TR/WCAG20/