Opportunities for Individual, Collaborative, & Community Accessibility on E-Learning Platforms

Reeti Mathur
Masters Student
Indiana University Purdue
University Indianapolis
Indianapolis, IN 46202, USA
remathur@umail.iu.edu

Erin Brady
Assistant Professor
Indiana University Purdue
University Indianapolis
Indianapolis, IN 46202, USA
brady@iupui.edu

Abstract
A number of modern tools for learning programming skills are online, and are used within classroom settings or at home for self-structured learning. Implementing appropriate web accessibility into these tools is crucial to allow students with visual impairments to learn these skills. However, many of these tools for early learners are visually based, leveraging "drag-and-drop" components that are not easily accessible through assistive technologies. This position paper talks in detail about the accessibility of secondary educational online systems – such as *Code.org's PlayLab* and MIT's *Scratch* – that are oriented for children. We describe how these portals could be made accessible for individual, collaborative, and community learning.

Introduction
There are many online portals designed to encourage children to think through technological problems creatively and collaboratively, such as Code.org's PlayLab and MIT Media Lab's Scratch project. Many of these tools are visual in nature, leveraging drag-and-drop components to simplify the process of learning to write pseudocode.

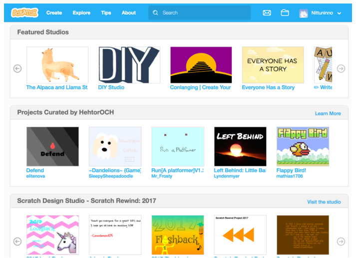

Figure 1: Screenshot of an inaccessible section of Scratch. The game icons do not have alt text. Voice Over reads each box with the image's file name. For example, the first image is described as "*link, 4028545"*.

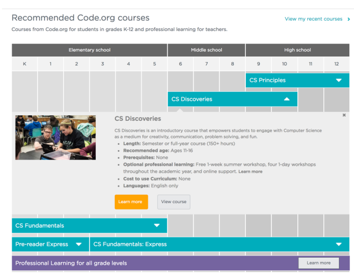

Figure 2: Screenshot of a tabular section of Code.org showing skill progressions. Because the table is not tagged appropriately, screenreaders cannot tell that there are drop-down tabs with details of each course. Semantic information on the schedule is also unavailable via screenreader.

These applications are often used by students in elementary school, where instructors direct children to create different designs, including games, stories, animations, and websites. Code.org has partnered with 180 of the largest school districts with a far-reaching total of 25,622,051 students[1], and Scratch has 24,156,577 projects shared, 20,096,805 users registered and 3,653,565 studios created[2].

While millions of children have learned from these educational sites, the visual content is rarely accessible or easy to use for blind and visually impaired children. According to the National Federation of the Blind (NFB), about 62,528 children are blind. Without better accessibility, students with disabilities may not be able to acquire the same technological skills as their sighted peers, and may need to depend on other sighted people, like instructors or parents, in order to interact with these e-learning sites. Encountering accessibility problems may lead to feelings of frustration or low self-efficacy among blind and visually impaired students.

Though these educational sites address accessibility to some extent, there are still many issues which make them inaccessible or difficult to interact with. Below, we quantify some of the accessibility problems on Code.org's PlayLab and Scratch that would impact blind and visually impaired students.

Inaccessibility of E-Learning Sites

We conducted automatic and manual accessibility testing to understand the accessibility of interactions on Code.org's PlayLab and MIT's Scratch, two of the most highly used drag-and-drop coding platforms. With both sites being visually based and heavily designed with graphics and User Interface (UI) elements in mind, we hypothesized that much of the content would be inaccessible, making it difficult for blind and visually impaired students to fully interact with it.

We performed an automated analysis of the home page of each site with the *WAVE Web Accessibility Evaluation Tool*[3] to identify sections of these sites which had obvious accessibility problems. This tool can identify images that lack alternative text descriptions, have inappropriate labels for form fields and links, or have missing or inappropriately structured headers. Many major accessibility issues arise due to the absence of alternative text descriptions ("alt text") for images, buttons and links [2], or the use of inappropriate alt text that misleads the user [1]. Text embedded in images also contribute towards inaccessibility, as screen readers are unable to recognize that text and read it aloud. We supplemented our automated analysis with manual evaluation and a test of the pages with the VoiceOver screenreader.

Overall Accessibility: The results of the WAVE evaluation are presented in Table 1 in the sidebar. In addition to missing alterative texts and form labels, there were structural and interaction-based issues.

Figure 2 demonstrates problems on Code.org. The coursework progression and programs for different grade levels are represented as a table in HTML, but are actually a GANNT-style chart that represents visual

[1] http://code.org/about

[2] https://wiki.scratch.mit.edu/wiki/Scratch_Statistics

[3] http://wave.webaim.org

MIT's Scratch: 45 Errors: 35 linked images missing alternative text, 7 empty or missing headings, 1 empty form label, I missing document language and 1 empty button 37 Alerts: 1 missing first level heading, 35 redundant links and 1 no-script element *Code.org:* 19 Errors: 5 missing alternative text, 2 linked images missing alternative text, 3 missing form labels, 1 missing document language, 8 empty links 5 Alerts: 1 skipped heading, 1 missing Audio/Video alternative text, 3 JavaScript jump menus

Table 1: Results of the WAVE accessibility evaluation.

information (e.g., the "CS Discoveries" program spans columns 7-11, indicating that it is appropriate for grades 6-10). This semantic information is not meaningfully annotated. The programs are additionally not tagged as interactable dropdown menus, meaning screenreader users may not be aware that they can expand each program for a description.

Authoring Tools: The drag-and-drop coding tools on both platforms are Flash-based, and do not contain accessibility information to be usable by screenreaders. As a technology, Flash is well-known for its accessibility problems, and the WebAIM group has stated that "…Flash accessibility is [probably] impossible due to it not being supported on many modern technologies, such as most mobile devices"[4]. The use of this standard means that authoring is inaccessible on the platforms. This follows the findings of Koushik and Kane [4] that traditional drag-and-drop coding tools are not accessible for people with visual impairments.

User-Generated Content: Figure 1 shows the extent of images without alt text on the Scratch homepage. The page shows icons of projects created with Scratch and uploaded by their creators. Because there is no alt text, the VoiceOver screenreader reads each image's file name, providing no information about the content of each image. These user-created projects make up a large amount of the material on the site and cannot be made accessible automatically.

Areas for Discussion at CHI 2018 Workshop

We want to understand a number of accessibility problems in the educational ecosystem of visually-based coding applications. While accessibility of the platforms must be improved to allow visually impaired students to interact directly with them, we must also examine the ways in which visually impaired students can collaborate with their sighted teachers and peers, and envision ways that these tools could be used to teach sighted students the basic principles for making their own content accessible when they collaboratively create and upload their work on such portals. Below, we discuss our potential research agenda, which we hope to refine alongside workshop attendees.

Individual Access: We described many common accessibility problems which need to be resolved for screenreader users to access the site. However, there are larger problems around the accessibility of these drag-and-drop coding tools which should be addressed to improve its **accessibility for individuals**.

These visual drag-and-drop tools make it extremely difficult for blind children to create projects on the sites. We want to evaluate alternatives for interacting with these components, such as voice-based interactive systems which produce audio output as in Blocks4All [5], or tangible interfaces like connecting blocks with tactile features as in Story Blocks [4].

We want to conduct contextual inquiries where we observe visually impaired students interacting with e-learning portals to help us gain further insight to their various accessibility problems. We also will build a browser extension to improve the accessibility of the authoring tools so we can perform our study.

[4] https://webaim.org/techniques/flash/

Collaborative Access: Teaching through online educational platforms can be done effectively in a collaborative environment, but there are common **collaborative access** issues which should be examined with these e-learning platforms. These issues may mirror some of the misconceptions that exist between blind employees and their sighted co-workers [3] – a sighted teacher or classmate of a blind student may prefer to interact with the content visually, meaning there must be easy translation between the visual and audio/tactile content.

We want to examine this by conducting studies of mixed-ability pairs of sighted and blind students, and sighted teachers with blind students, to examine how they can collaborate over this content in different mediums. This work would be conducted through our connections with the Indiana School for the Blind and Visually Impaired, which hosts residential students but also does assistive technology outreach in local schools.

Community Access: Finally, we recognize that many of the accessibility problems on these sites are the result of them hosting *user-generated content*. Regardless of the accessibility improvements implemented by the site, we must also build a **community knowledge of accessibility** among the sighted creators who upload and share their work.

The platform could be augmented through the mandatory inclusion of "alt text" descriptions for each character or user-created image in the code. At schools, sighted peers and instructors of visually impaired kids could help improve the accessibility of such online portals by making it a regular practice to caption their work and provide a detailed description at every level of the creation process, using this process as an opportunity to be introduced to web accessibility principles. We would like to test this process with content creators to rate the quality of their descriptions or captions.

Conclusion

We hope to take advantage of this CHI 2018 Workshop to get feedback on our research agendas, and to plan a study of e-learning site use by children with visual impairments and their sighted teachers and peers. The valuable feedback and interaction with the other participants and organizers of this workshop would be valuable to guiding our future study.

References

1. Chieko Asakawa. 2005. What's the web like if you can't see it? In *Proceedings of W4A 2005*, 1–8.
2. Jeffrey P. Bigham, Irene Lin, and Saiph Savage. 2017. The Effects of "Not Knowing What You Don`t Know" on Web Accessibility for Blind Web Users. In *Proceedings of ASSETS '17*.
3. Stacy M Branham and Shaun K Kane. 2015. The Invisible Work of Accessibility: How Blind Employees Manage Accessibility in Mixed-Ability Workplaces. In *Proceedings of ASSETS '15*, 163–171.
4. Varsha Koushik and Shaun K Kane. 2017. Tangibles + Programming + Audio Stories = Fun. In *Proceedings of ASSETS '17*, 341–342.
5. Lauren R Milne, Catherine M Baker, and Richard E Ladner. 2017. Blocks4All Demonstration: a Blocks-Based Programming Environment for Blind Children. In *Proceedings of ASSETS '17*, 313–314.