Web Based Accessible Computer Science Applications for K-12 Students with Disabilities

Daniela Marghitu, Auburn University,
marghda@auburn.edu

Meenakshi Das, Auburn University,
mzd0107@auburn.edu

Speakers

+ Daniela Marghitu

Dr. Daniela Marghitu received her B.S. in Automation and Computing from Polytechnic University of Bucharest, and her Ph.D. degree in Automation and Computing from University of Craiova. She is a faculty member in the Computer Science and Software Engineering Department at Auburn University, where she has worked since 1996. She is the Co-PI of NSF CISE "EAGER: An Accessible Coding Curriculum for Engaging Underserved Students with Special Needs in Afterschool Programs". For more info see her official site http://www.eng.auburn.edu/~daniela/.

+ Meenakshi Das

Meenakshi 'Meena' Das is a masters student studying computer science at Auburn University, where she is a Graduate Research Assistant working on accessible computing education under Dr. Daniela Marghitu. This work has resulted in two publications at the International Conference on Human-Computer Interaction, and she was also named an NCWIT Collegiate Award Finalist.

- According to the US Bureau of Labor Statistics, the employment of computer and information research scientists is projected to grow **15 percent from 2019 to 2029**, much faster than the average for other occupations.
- However, **individuals with disabilities are less likely** to pursue pursue science, technology, engineering, and mathematics degrees.

Why? Inaccessible multimodal interfaces

Screen-reader or Text-to-Speech for Blind/Low Vision Individuals.

Speech-to-Text for motorically disabled individuals.

American Sign Language (ASL) for Deaf/Hard of Hearing Individuals.

Why? Limited multimodal learning strategies.

Using Images, Graphics illustrations, and unplugged activities.

Tactile materials for Blind users.

Universal Design for Learning.

Why? Inaccessible Instruction.

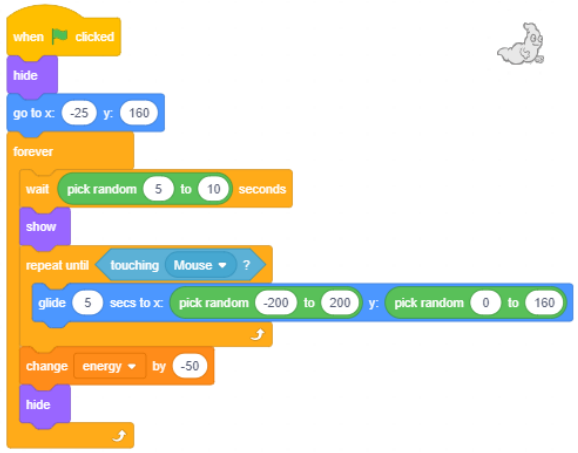
- Inaccessible documents.
- Uncaptioned teaching videos.
- Low expectations for students with disabilities.

Why use web-based apps to teach programming to K-12 students with disabilities.

+ Most popular platforms to teach coding to kids, such as block-based programming, are web-based.
+ No time spent on installation, can jump right onto the learning.
+ Can access through mobile, tab or a laptop.
+ Cross-platform compatibility, saving progress on cloud.
+ Web Content Accessibility Guidelines (WCAG) are well-established.

Alas, not all are accessible.

Block-Based Programming



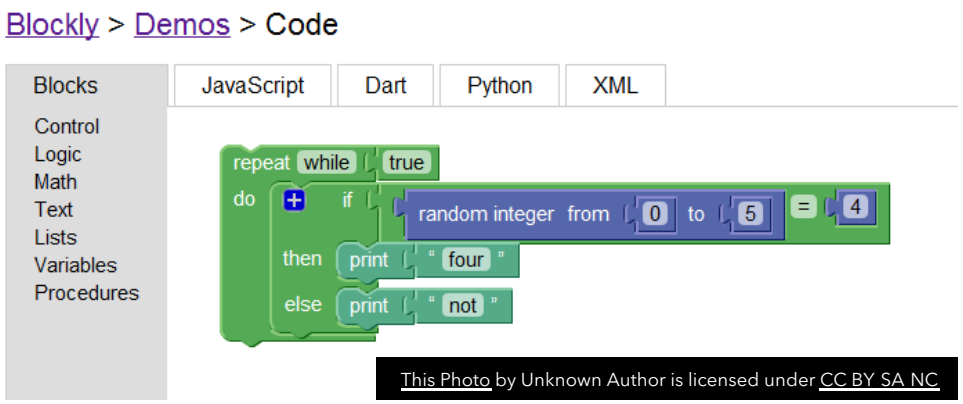
Applications such as App Inventor and MIT Scratch, commonly used to teach computer programming to K-12 students.

They depend heavily on visual cues and mouse input for drag and drop – which make them inaccessible.

Making Block-Based Programming Accessible to Blind Individuals.

Related Work

+ Google first released their accessible version of Blockly which replaced the drag and drop layout with a text layout for screen- reader compatibility.
+ Researcher Stephanie Ludi's work mainly focused on adding screen reader capability to Blockly, along with key-board navigation, while maintaining its original block-based interface [1].
+ Google later re-released their own version of Blockly with Keyboard navigation for its blocks.



Why Blockly?

- Web-Based – no heavy installations necessary!
- Client Side library in JavaScript – no server-side dependencies.
- Popular block-based apps(Scratch, Code.org etc.) use Blockly as their base.

Making Block-Based Programming Accessible to Blind Individuals.

Our Solution

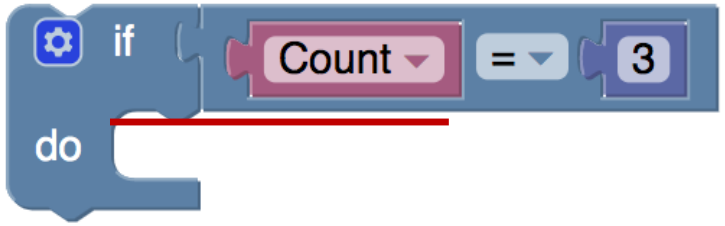
Built upon Keyboard Navigation Feature by Blockly.

Both screen reader and self-voicing solution. Screen-readers are expensive and have an immense learning curve for K-12 Students.

Retain Spatial Output. “Access techniques that distort or remove spatial information may reduce users’ spatial understanding and memory” [2].

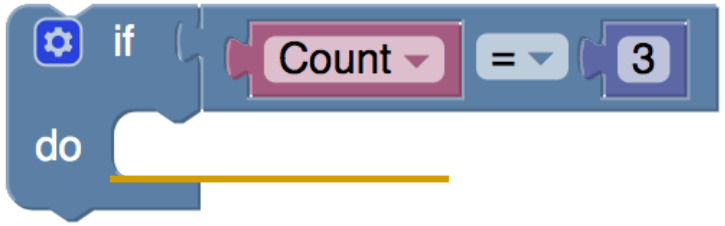
Auditory Cues: Binaural Spatialization to assist students to understand the opening and closing of nested code blocks.

Binaural Spatialization.



Web libraries such as Web Speech API and Web Audio API make it very convenient to implement these features and work on all major browsers.

Binaural Spatialization.



Web libraries such as Web Speech API and Web Audio API make it very convenient to implement these features and work on all major browsers.

Demo: Text to Speech

+ Watch the Demo Here: https://youtu.be/vlPxnPPrz9o

Evaluation Results

+ **Improved Audio Feedback:** Make granular aspects accessible.
+ **Personalization:** Allow features to control speed and verbosity of text to speech.
+ **Screen-Reader Functionality**: For more adult users proficient with a screen-reader.
+ **Improved Cursor:** Add functionality to access connections easily.

Demo: Using Screen-Reader

+ Watch the Demo Here: https://youtu.be/QuumG0n3K2M

Text-to-Speech vs Screen-Reader

- Low Vision High-School Student with low screen-reader proficiency preferred TTS as screen-readers have a learning curve.
- Useful when screen-reader software is not present.

- Blind Adult PhD student preferred screen-reader.
- Experienced screen-reader professionals are more comfortable using screen-reader. With TTS, they have to turn screen-reader off.

We believe screen-reader proficiency should not be prerequisite to learning to code, hence we developed both TTS and screen-reader approaches – **multimodal interface.**

Making Block-Based Programming Accessible to those with Motor Disabilities.

- Individuals with motor disabilities (e.g. hand, shoulder) have difficulties operating the Keyboard. They may rely on input modalities such as speech, eye-gaze, or switch control.
- Programming using voice (Speech-to-Text) is becoming popular these days.

Making Block-Based Programming Accessible to those with Motor Disabilities.

Related Work

- Myna, a tool developed in Java runs parallel to Scratch as a voice recognition system [3].
- When Myna is started and Scratch is opened, users can create programs within Scratch solely through voice.

Making Block-Based Programming Accessible to those with Motor Disabilities.

Our Solution

Built upon Keyboard Navigation Feature by Blockly.

Speech-to-Text JavaScript Library *AnnYang*.

Natural Language Processing using client-side JavaScript library *compromise*.

Dispatch Keyboard events and keep track of various states using a finite state machine.

Making Block-Based Programming Accessible to those with Motor Disabilities.

Initial Design

+ Voice command mapped to a single Keyboard Event.
+ Quite time-consuming and adds cognitive load on the user.
+ Let's look at a demo.

Command	Action	Mapped Key
Toolbox	Enters the Toolbox	T
in	cursor proceeds to next block element	D
out	cursor proceed to previous block element	A
next	cursor proceed to next line	S
previous	cursor proceed to previous line	W
select	selects a block from the toolbox	Enter
mark	marks a connection on a workspace block	Enter
insert	inserts selected block to a marked connection	I
detach	detaches a block from another block	X
tooltip	displays the tooltip for a selected block	CTRL + T
delete	deletes a block	DELETE
information	displays text representation of a block	CTRL + I

Demo: Speech to Text

+ Watch the Demo Here: https://youtu.be/cifl1w1Ggl8

Making Block-Based Programming Accessible to those with Motor Disabilities.

Improved system using NLP

+ Voice command mapped to ***multiple*** Keyboard Events.
+ **Reduces** cognitive load on the user.
+ Let’s look at a demo.

Demo – Using NLP

+ Watch the Demo Here: https://youtu.be/ikU6DQSMxGU

Benefits of Natural Language Understanding

Fitts' law states that the **"amount of time required for a person to move a pointer (e.g., mouse cursor) to a target area is a function of the distance to the target divided by the size of the target. Thus, the longer the distance and the smaller the target's size, the longer it takes" [4].**

+ Reduces cognitive load on user.
+ Saves time to carry out tasks.

NLP model can be trained on variety of voices to improve recognition.

Web libraries such as *AnnYang* for speech recognition and *compromise* for natural language processing make it very convenient to implement these features and work on all major browsers.

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

- Deaf individuals often **acquire sign language as their first language**, and they are most fluent and comfortable in it [5].
- **Half of deaf students** pass from secondary school with a fourth-grade reading level or less [6].
- the frequently reported low literacy levels among students with severe hearing disability were partly due to the discrepancy between their ***"incomplete spoken language system and the demands of reading a speech-based system" [7].***

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

Related Work

+ Apple developed eight computer science concept videos in ASL.
+ ASLCORE- developed STEM technical terminology in ASL, so deaf individuals do not have to depend always on fingerspelling.

Pre-lingual individuals *(born with deafness or became deaf very young) who learned ASL as their first language will more likely be interested to learn coding through ASL.*

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

App feature considerations: Based on heuristic evaluation for Deaf Web User Experience (HE4DWUX) [8].

This heuristic evaluation is based on the well-known usability heuristics developed by Jakob Nielsen. The HE4DWUX consists of similar heuristics as Nielson's and provides additional context with regards to accessibility for the D/HH.

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

1. Sign Language Video Content – We filmed computer science concept videos in ASL.

Demo: ASL

+ Watch the Demo Here: https://youtu.be/bpysLmkSi_s

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

2. Visibility of status and actions – Not limited to audio cues, using text with icons.

3. Match between website and the world of the Deaf–The website should be aligned with language and concepts from Deaf culture:

A Native ASL speaker starred in our videos to ensure authenticity. We also partnered with the non-profit *Deaf Kids Code*.

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

4. User Control and Freedom: ASL signs for every drag and droppable ‘block’ like “IF-ELSE” or “WHILE”, are displayed as tooltips for a block. Users can switch between ASL and Text.

Demo: Tooltips

+ Watch the Demo Here: https://youtu.be/0CV5herpHsU

Making Block-Based Programming Accessible to Deaf/Hard of Hearing Individuals.

5. Captions– "Captioning is perfect for the post-lingual deaf or hard-of hearing audience; it presents content in an accessible format, in the primary language of the user"

6. Aesthetic and minimalist design - Minimize distracting and cluttering content.

7. Personal skills – "Graphics, pictures, videos, and the different forms of multimedia can facilitate learning of complex theories and generalizations, which are usually challenging for deaf students that do not know how to use sign language" [9].

Conclusion

Provide multimodal accessible interfaces to learn.

Provide accessible curriculum to learn.

Use the power of web!

And last but not the least – remember **Accessibility is a right!**

Acknowledgements

This work has been made possible due to **NSF Grant - EAGER: An Accessible Coding Curriculum for Engaging Underserved Students with Special Needs in Afterschool Programs #1842092** and a **SIGCSE Special Projects Grant.**

References

1. Ludi, Stephanie, and Mary Spencer. "Design Considerations to Increase Block-basedLanguage Accessibility for Blind Programmers Via Blockly." Journal of Visual Languages and Sentient Systems 3.1 (2017): 119-124.
2. Kane, S. K., Morris, M. R., Perkins, A. Z., Wigdor, D., Ladner, R. E., & Wobbrock,J. O. (2011, October). Access overlays: improving non-visual access to large touch screens for blind users. In Proceedings of the 24th annual ACM symposium on User interface software and technology (pp. 273-282).
3. Wagner, Amber, Ramaraju Rudraraju, Srinivasa Datla, Avishek Banerjee, Mandar Sudame, and Jeff Gray. "Programming by voice: A hands-free approach for motorically challenged children." In CHI'12 Extended Abstracts on Human Factors in Computing Systems, pp. 2087-2092. 2012.
4. "What Is Fitts' Law?" The Interaction Design Foundation, https://www.interaction-design.org/literature/topics/fitts-law. Accessed 23 Apr. 2021.
5. Huenerfauth M., Hanson V.: Sign language in the interface: Access for deaf signers. In C Stephanidis, editor, The Universal Access Handbook, CRC Press, chapter 38, pp. 1–18, (2009).
6. Traxler, C.: The Stanford Achievement Test, 9th Edition: National Norming and Performance Standards for Deaf and Hard-of-Hearing Students., Journal of Deaf Studies and Deaf Education 5(4), pp. 337-48, (2000).
7. Geers, A.: Spoken Language in Children with Cochlear Implants. Advances in Spoken Language Development of Deaf and Hard of Hearing Children–Spencer P, Marschark M, eds. New York: Oxford University Press, pp. 244–270, (2006).
8. Yeratziotis. A., Zaphiris. P.: "A Heuristic Evaluation for Deaf Web User Experience (HE4DWUX)," International Journal of Human– Computer Interaction 34(3), pp. 195-217, (2018).
9. Kourbetis, V., Boukouras, K., and Gelastopoulou, M.: Multimodal accessibility for deaf students using interactive video, digital repository and hybrid books. Proceedings of the International Conference on Universal Access in Human-Computer Interaction. Toronto, Canada, pp. 93-102, (2016).