Sonification Blocks: A Block-Based Programming Environment For Embodied Data Sonification

Jack Atherton
Center for Computer Research in Music and Acoustics
Stanford University
Stanford, CA, USA
lja@ccrma.stanford.edu

Paulo Blikstein
Graduate School of Education
Stanford University
Stanford, CA, USA
paulob@stanford.edu

IDC '17, June 27-30, 2017, Stanford, CA, USA

ACM ISBN 978-1-4503-4921-5/17/06.
http://dx.doi.org/10.1145/3078072.3091992

Abstract
High school students often struggle to find the motivation to learn to program. Music can be a powerful motivator for these students, but existing tools that combine music production with programming often fail to meaningfully engage students with core computer science concepts. Sonification Blocks was created to shift the focus back toward big ideas in programming. Sonification Blocks is a programming language for data sonification, the process of creating audio algorithms and controlling them with streams of data. Its implementation as a block-based language with clear, interactive data visualizers allows high-school-aged learners to develop computational literacy. Furthermore, the act of manipulating sound parameters with data streams that are controlled through body motion may help connect learners with powerful ideas in programming and data science.

Author Keywords
computer science education; music; sonification; embodied cognition; constructionism

ACM Classification Keywords
K.3.2 [Computers and Education]: Computer and Information Science Education; H.5.5 [Information Interfaces and Presentation (e.g., HCI)]: Sound and Music Computing

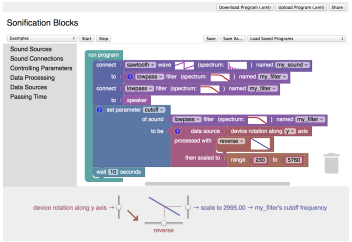

Figure 1: The Sonification Blocks workspace. Sonification Blocks is implemented in HTML, CSS, and JavaScript using Google Blockly and Web Audio [5, 8]. As such, it is cross-platform and free to use in any modern browser. Learners can save programs in the local browser storage, download and upload programs as XML files, and share web links that recreate their workspace when opened. These features allow learners to share their creations publicly and to easily collaborate with peers.

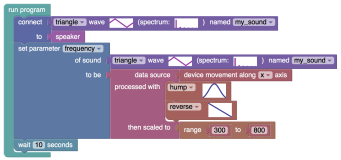

Figure 2: A program written in Sonification Blocks, sonifying the device's linear acceleration along the x axis by connecting it to the frequency of a triangle wave oscillator. Running this program also shows the live data processing visualization of Figure 5.

Introduction

Sonification Blocks (https://ccrma.stanford.edu/~lja/sonification/) is a web-based learning environment designed to introduce high-school-aged learners to introductory concepts in programming, data processing, and computer sound production. Data sonification is the act of controlling the parameters of an audio algorithm with values from a stream of data for the purpose of learning about the data. With Sonification Blocks, learners construct audio algorithms in a block-based programming language, controlling parameters of the sound by using sensors built into the device on which they program and by shaping the output of those sensors with a variety of "atomic" data processing functions (*atoms*). Our hope is that learners using Sonification Blocks will make progress toward two key components of computational literacy: object oriented thinking, especially in the domain of sound, and the processing and connection of flows of data.

Our work is grounded in the constructionist tradition, in that students learn by building and sharing personally meaningful sonifications [1]. Sonification Blocks also draws on ideas from embodied cognition in allowing the application of body-based cognition to a non-situated activity [10]. Playing a physical instrument is a situated activity, as musicians can feel their instrument's form and vibrations (the methods of generating sound) as they play. Audio programming divorces the thoughts associated with making sound from the production of sound, both in time (first write a program, then run it) and in physicality (the sound mechanism is hidden inside a device and is not controlled by bodily actions). The use of physical sensors for the control of sonification programs allows learners to apply bodily intuition for motion to the relatively non-situated task of audio programming, and the continual recompilation that occurs whenever the code is changed reconnects thought and audio output in time.

Background and Related Work

Introductory computer science students often struggle to stay motivated in text-only paradigms [7]. Some researchers have turned to music and audio programming as a motivator that works well across diversity of gender, race, and class [6, 7, 9]. A basic approach simply introduces the ability to play back or *trigger* audio files programmatically [2]. This ability is technically sufficient for some applications like games, but does not invite introspection about the act of sound generation itself. Tellingly, students who engage in sound triggering activities like making music videos engage less with core programming concepts like variables and control flow when compared to students creating games [2].

Two promising tools that use audio programming for computer science education are *EarSketch* and *BlockyTalky*. *EarSketch* is a "computational music remixing" library used to motivate high school students with no music background through a comprehensive computer science curriculum [6]. However, its lack of physical interaction via sensors or controllers prevents the use of bodily understanding during the music-making process. *BlockyTalky* is a block-based programming language and toolkit for creating sensor-controlled synthesizers that can communicate with each other over the internet [9]. These devices helped children aged 11–14 learn about advanced computer science topics, but long delays between sensor readings and audio output diminished students' ability to interpret how their programs were working. Also, the focus on creating music for a public performance led many students to mimic pop music rather than engaging with programming concepts.

In the realm of data science, the *Scratch Community Blocks* extension motivated children aged 11–15 to start building a data science literacy by allowing them to construct personally meaningful visualizations of data about the Scratch

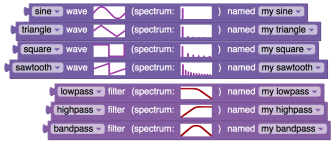

Figure 3: All variants of waveform blocks and effect blocks. Each of these blocks can be set to one of several sub-types, such as *triangle wave* or *lowpass filter*.

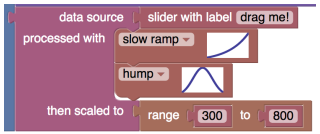

Figure 4: The data processing blocks. Here, a slider labeled "drag me" is fed through an exponential atom, followed by a gaussian atom, then scaled up. Figures 6 and 7 show the live updating graphs that are shown when this block is run.

Figure 5: The visualization of a data processing chain. Here, a data stream is sent as an argument to two data processing atom functions composed together, then scaled up to a larger value. The sliders, graphs, and scaled value all update in real time, and each slider can be moved manually to determine its impact on the rest of the data processing chain and on the audio algorithm.

ecosystem [4]. However, it limited the notion of "data" to static information accessed through a database. Notably, users expected to be able to update visualizations in real time as the world changed, which was not possible due to the project's slow query times and caching system.

Design Overview

Sonification Blocks focuses on the activity of data sonification rather than the activity of making music. We were also intentional in choosing the layer of technical complexity to expose to students ("selective exposure," [3]). Data sonification allows users to engage with data in an immediate, time-varying way and to engage in programming sound without the distractions of composing or copying music. The top-level blocks of the language allow learners to connect audio objects to each other, to set parameters of those objects, and to pass time. Importantly, parameters can be set via the output of other audio objects and via the output of a data source, allowing complex audio signal chain configurations and enabling the sonification aspect of the language.

Audio Blocks

The two audio object blocks are the *waveform* and *effect* blocks (Figure 3). Each block shows the frequency domain representation of the underlying audio object so that its effect on the sound can be understood visually and aurally.

When one of these blocks is attached to a *set parameter* block, the parameter block updates its dropdown menu to show the parameters specific to the attached block. This behavior enables progress toward the set of learning goals that compose object-oriented thinking by instilling two notions: first, that objects have properties that are held in common with similar objects but not with all objects; second, that different object types belonging to the same larger category can be used interchangeably in some contexts.

Data Processing Blocks

The data blocks are linked together with the data processing chain block (Figure 4). This block can be used in the parameter block instead of a number or audio object. It takes a live data source, shapes it according to one or more data processing atoms, then scales the result to a specified range and sets the parameter with that value.

When a program is run, a live-updating chain of graphs and sliders representing the data processing flow of the program appears at the bottom of the page (Figure 5). Each slider can be moved manually to affect the rest of the chain, allowing debugging at a fine-grained level. If more than one data processing atom is linked together, the graph of composing each atom's function together is shown (Figure 6). This graph can be broken apart into the constituent atoms to observe the effect of each individually (Figure 7).

Sonification Blocks currently includes two kinds of data sources. Slider data sources let learners control the data flow by manually manipulating a slider located next to the visualizer chain, allowing them to explore the effect of each part of the chain at their own pace. Alternatively, when using a mobile device equipped with an accelerometer, learners can use the angular position, linear acceleration, or rotational acceleration of their device as a data source. Using accelerometer data sources allows learners to understand their audio algorithms in an embodied way by manipulating the algorithms with subtle hand and body movements.

Live Feedback

Changes in data sources propagate to parameters in the audio algorithm without any perceivable delay. This low latency allows learners to readily interpret the effect their gestures have on their audio algorithm. Also, a running audio algorithm is updated immediately whenever the underlying program is edited, affording quick back-and-forth testing.

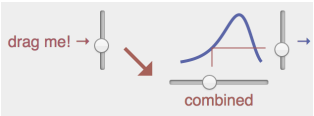

Figure 6: A combination graph, showing an exponential atom composed with a gaussian atom.

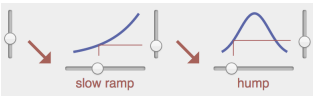

Figure 7: The individual atom component graphs, broken apart from the graph of Figure 6.

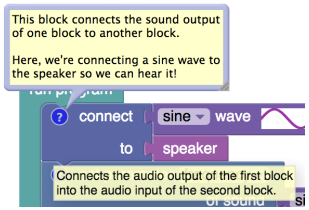

Figure 8: An example program comment (upper) and a mouse hover tooltip (lower).

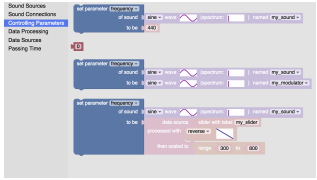

Figure 9: Shadow blocks in the toolbox, suggesting three different possible uses of the same block.

Scaffolding

Each block has a tooltip that explains what the block does whenever the cursor is hovered over it. Additionally, learners may load a number of example programs, which increase slowly through topic complexity and program length and have comments to explain how each unfamiliar block is used (Figure 8). Within the toolbox menus where learners find blocks for their programs, the inputs to complex blocks are filled out with potential configurations using *shadow* blocks. These blocks are lighter in color, suggesting a partial solution for how to use the parent blocks while also allowing learners to fill in the inputs with their own blocks. If a block can be used in multiple ways, it appears multiple times in the toolbox with different shadow inputs to encourage learners to explore all the options (Figure 9).

Conclusion

Sonification Blocks is a block-based programming language for data sonification. It allows learners to program audio synthesis algorithms and control them with streams of data, encountering big ideas in object-oriented thinking and computer music. Clear, interactive data visualizers show the effects of atomic data processors, allowing learners to start building a data science literacy. When using accelerometer data sources, learners can translate their bodily movements into sound for an embodied audio programming experience. Sonification Blocks provides a context and a language for exposing students to powerful ideas in both programming and sound synthesis.

Acknowledgments

We thank Richard Davis, Engin Bumbacher, and Chris Proctor for their invaluable feedback on early prototypes and on the manuscript. We also thank the Lemann Center at Stanford for its support.

References

[1] Edith Ackermann. 2001. Piaget's constructivism, Papert's constructionism: What's the difference?. In *Constructivism: Uses and Perspectives in Education*.

[2] Joel C. Adams and Andrew R. Webster. 2012. What do students learn about programming from game, music video, and storytelling projects?. In *Proceedings of the 43rd SIGCSE Technical Symposium*. 643–648.

[3] Paulo Blikstein. 2015. Computationally Enhanced Toolkits for Children: Historical Review and a Framework for Future Design. *Foundations and Trends in Human-Computer Interaction* 9, 1 (2015), 1–68.

[4] Sayamindu Dasgupta and Benjamin M. Hill. 2017. Scratch Community Blocks: Supporting Children as Data Scientists. *Preprint, arXiv:1702.00112* (2017).

[5] Google Developers. 2017. Blockly. (8 March 2017). Retrieved from https://developers.google.com/blockly/.

[6] Jason Freeman, Brian Magerko, Tom McKlin, and others. 2014. Engaging underrepresented groups in high school introductory computing through computational remixing with EarSketch. In *Proceedings of the 45th SIGCSE Technical Symposium*. 85–90.

[7] Mark Guzdial and Elliot Soloway. 2002. Teaching the Nintendo generation to program. *Commun. ACM* 45, 4 (2002), 17–21.

[8] Mozilla Developers Network. 2017. Web Audio API. (8 March 2017). Retrieved from https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API.

[9] Benjamin R. Shapiro, Annie Kelly, Matthew Ahrens, and Rebecca Fiebrink. 2016. BlockyTalky: A Physical and Distributed Computer Music Toolkit for Kids. In *Proceedings of the 2016 International Conference on New Interfaces for Musical Expression (NIME16)*.

[10] Margaret Wilson. 2002. Six views of embodied cognition. *Psychonomic bulletin & review* 9, 4 (2002).