H. Lindetorp and K. Falkenberg, "Evaluating Web Audio for Learning, Accessibility, and Distribution"
J. Audio Eng. Soc., vol. 70, no. 11, pp. 951–961, (2022 November).
DOI: https://doi.org/10.17743/jaes.2022.0031

Evaluating Web Audio for Learning, Accessibility, and Distribution

HANS LINDETORP,[1, 2] **AND KJETIL FALKENBERG**[2]
(hans.lindetorp@kmh.se) (kjetil@kth.se)

[1]*Department of Music Production, Royal College of Music, Stockholm, Sweden*
[2]*Sound and Music Computing Group, KTH Royal Institute of Technology, Stockholm, Sweden*

Web Audio has a great potential for interactive audio content in which an open standard and easy integration with other web-based tools makes it particularly interesting. From earlier studies, obstacles for students to materialize creative ideas through programming were identified; focus shifted from artistic ambition to solving technical issues. This study builds upon 20 years of experience from teaching sound and music computing and evaluates how Web Audio contributes to the learning experience. Data was collected from different student projects through analysis of source code, reflective texts, group discussions, and online self-evaluation forms. The result indicates that Web Audio serves well as a learning platform and that an XML abstraction of the API helped the students to stay focused on the artistic output. It is also concluded that an online tool can reduce the time for getting started with Web Audio to less than 1 h. Although many obstacles have been successfully removed, the authors argue that there is still a great potential for new online tools targeting audio application development in which the accessibility and sharing features contribute to an even better learning experience.

0 INTRODUCTION

There is a great potential for web technologies and in particular Web Audio API [1] as creative platforms for learning audio and music [2] and programming [3, 4]. There are many examples of online tools for working with audio in the browser targeting, e.g., music production (Sound Trap, www.soundtrap.com), ear-training and music learning (Music First, https://www.musicfirst.com), and experimental applications (Nü Soundworks, https://ircam-cosima.github.io/soundworks-nu).

The sound and music computing community has seen a continued development in open-source languages and standards that contribute to accessible environments for creation across platforms and devices, which in turn allow for producing shareable content. With web technologies, the environments do not even need installation or software updates if the standard does not change, and there is a rapidly growing support for microphones, sensors, and cameras as input sources, which makes the platform interesting for a wide range of audio applications. The authors argue that there are many established practices to develop further and new areas to explore with web technologies in music production, sonification, new expressive interfaces, and more. These align with the curricula of artistic and sound and music computing–related educations.

In their teaching in music production and music technology, the authors encounter mixed cohorts of students with varying experience of programming in frameworks like Pure Data and SuperCollider, and practical production work tools like digital audio workstations. Recent studies indicate that there are still high thresholds to enter developing interactive audio and music applications, partly due to little programming experience [5, 6]. The authors have observed that this obstacle has a negative impact on student motivation and causes lowered artistic ambitions in their projects [7]. A main challenge the authors experience with the traditional software platforms is that they do not allow for students to collaborate, distribute, and share their work without some additional effort.

To remove obstacles and tackle the challenges the authors have experienced, Lindetorp developed WebAudioXML (waxml) [8, 9], an open-source framework[1] that offers an XML syntax for configuring audio objects and mapping user interactions to audio parameters. It uses a declarative approach to describe the audio objects, like how HTML represents visual content in a web page. Waxml is an abstraction of the Web Audio API and a platform for designing and exploring an accessible coding language for creative

[1] https://github.com/hanslindetorp/WebAudioXML.

artists and technical developers. The term "accessibility" has many interpretations; in music technology contexts, the term is used to mean a coding environment that is open, well-supported, free, and easy to obtain, understand, and use, which makes it accessible for learning, being creative, and distributing productions.

The overall aim with the waxml project is to open more possibilities for producers of creative or artistic content to create interactive audio applications without having to learn several layers of coding languages and technologies. The authors seek solutions that help producers retain their creativity and stay focused on the artistic expression rather than solving technical issues, and the authors want to contribute to formats that are accessible and easily distributed. This study evaluates Web Audio using waxml in a pedagogical context against the background of having examined more than 80 different student projects at three universities.

The evaluations included here coincide with two moments in waxml's development and involve student projects and workshops at The Royal College of Music (KMH) and KTH Royal Institute of Technology (KTH). The aim of the first evaluation, conducted during COVID-19 lockdown, was to explore the creation of interactive audio and music applications and share the result between students without having to meet physically [10]. Results showed that students performed better compared with previous course rounds and pointed toward possibilities for further lowering the thresholds for artistic work and removing practical barriers for ambition.

The second evaluation builds on the long experience of artistic projects involving programming with waxml, reviews whether the increased accessibility to coding improved the students' learning experiences, and investigates effects of having shorter sessions and project periods. The task was to build a gesture-controlled musical instrument using waxml and the MediaPipe library [11] with a primary focus on mapping gestures to basic synth features like pitch, filter frequencies, volume, panning, and triggering of samples and envelopes.

The following section describes the teaching methods, how research and education are integrated, related work, and the authors' prior experience. SEC. 2 presents waxml with its most relevant components for this study. SEC. 3 describes the data collection and methods. SEC. 4 summarizes the output from the student projects and results of the evaluation. SEC. 5 discusses the results, and, finally, SEC. 6 draws some conclusions with implications for the future.

1 BACKGROUND

Since its release in early 2020, waxml has been used in 13 courses at KMH (http://www.kmh.se), KTH (http://www.kth.se), and Södertörn University (http://www.sh.se), all of which are in Stockholm, Sweden. It has become the authors' main platform for work with sound interaction and interactive music, with artistic, educational, and research ambitions, and includes tools for interaction, such as creative variable mappings and output of statistical user data. In addition to educational activities, waxml is regularly used in research and artistic productions at KMH and KTH.

This section describes how this research is integrated with ongoing education and how the key values of this work feed into the courses. It also describes related work using similar technologies and briefly lays out important stages of the development of waxml leading up to the current study.

1.1 Teaching Methods

At KMH and KTH, the authors teach sound and music computing and production using a problem-based, learner-centered approach [12]. The authors often introduce new technology through workshops and seminars and then supervise projects for which the students explore ideas and solutions to achieve artistic or design goals. After the practical work, they report, evaluate, and reflect on the result and give feedback on the process. The texts are assessed against the course's learning outcomes.

1.2 Pedagogical Values

In their curricula, the authors are obliged to include learning objectives relating to ethical values, and the students are expected to reflect critically on aspects like inclusion, accessibility, and equality with a broad perspective. This urges the authors to suggest technologies based on open standards and equipment available to all students regardless of, e.g., financial situation, gender, ethnicity, or artistic practice. The W3C Mission Statement [13] overlaps to a great extent with those values and is one reason why the authors want to contribute to web technologies and Web Audio with knowledge, standards, applications, and systems for offering more creative sound and music computing learning experiences.

Another value that directs the authors' efforts is that they want the courses to be focused on the core learning objectives. Particularly, they want the artistic and design-oriented courses to be pertinent to the sound, music, and computing aspects; irrelevant technical struggles and obstacles cause students to lose focus and interest. The authors also value a collaborative learning environment where work can easily be shared between students and supervisors.

1.3 Related Work

There is a growing number of research projects in the Web Audio community that aim at meeting needs and solving problems related to the work presented in this study. There are several JavaScript abstractions of Web Audio API that simplify the creation of the Audio Graph and add custom objects that emulate audio effects and synthesizers, e.g., WAAX [14] and Tone.js [15]. There are also graphical abstractions like JSPatcher [16], [17], and Quint.js [18] and online coding environments like BRAID [19] and EarSketch [20] aiming at making audio development accessible for creators with little or no prior programming experience. Compared to these frameworks, waxml might not be as robust as some of the JavaScript abstractions and not as user-friendly as the graphical interfaces but shares similarities with X3D Audio API [21] in that they both have a descriptive and extensible syntax that is easy to read and understand.

None of the most common programming environments within the sound and music computing field—CSound, Max/MSP, Pure Data, and SuperCollider—were initially intended to be used for web applications because they were developed long before web browsers were capable of music programming [22]. In recent years, there have been several attempts made to integrate these environments in browsers, such as Flocking [23], WebPd [24], Gibberish/Interface [25], and PNaCl [26].

Finally, there are projects that use several components to build systems and applications that share similarities with the ones developed for this study, like interactive audio networking systems [27, 28] and real-time sonification of body movements [29].

1.4 Past Experiences

For more than 20 years, the authors have used different platforms for developing interactive sound and music applications at KMH and KTH, including Pure data, SuperCollider, Macromedia Director, and Adobe Flash. Since the introduction of Web Audio API, the authors have developed methods for their students to implement content more easily for interactive environments, and these projects have contributed to the design and development of waxml. Typically, students request means and features for realizing artistic goals, and through workshops and discussions, these features are then being implemented, used, and evaluated. Below is a summary of some of these projects that have contributed to the development of waxml.

1.4.1 Adaptive Music in Museum Exhibitions

Fourteen Master's students from KMH interpreted the Nobel Prize in an interactive exhibition at the Nobel Prize Museum in Stockholm using infrared sensors, buttons, and touch screens, which controlled an adaptive music composition that reflected the activity among the visitors [30]. The project proved Web Audio to be a robust platform for playing back thousands of audio files in a multi-channel speaker setup with perfect synchronization for several months. The framework used for the audio playback was iMusic,[2] a predecessor to waxml developed in 2014.

1.4.2 Multi-Sensory Music Installation

Students and researchers from KTH and KMH have designed, built, composed for, and evaluated a large-scale, multi-sensory, inclusive digital instrument called the "Sound Forest" at the Swedish Museum of Performing Arts [31, 32]. Web Audio was used to map the visitors' interactions with the instrument to sound and haptic feedback [33].

1.4.3 Multi-Channel Audience-Controlled Music

Bachelor's students at KMH produced interactive compositions for a multi-channel super-surround speaker setup called "Klangkupolen" [34], in which the audience participated by controlling the playback using gestures [7]. The setup was built with a Node.js server [35], socket.IO [36], and Web Audio and proved to be a useful platform for multi-user music interactions. The study also identified technical barriers for some students, in which the steep learning curve for coding in JavaScript prevented them from fully engaging in the creative process. The insights from this study were important for initiating the development of waxml.

1.4.4 Sonification of Data

Ten students without any prior experience of audio programming participated in a study [37] in which the WebAudioXML Sonification Toolkit was evaluated as an interface for a browser-based sonification process. It was concluded that the toolkit was accessible with a low threshold for creating interactive sonifications and that the interface would benefit from making mapping data to audio parameters even simpler, to have features for preprocessing data, more audio generators, and more mixing features like a digital audio workstation; these were implemented in the subsequent version [9].

2 WEBAUDIOXML

Waxml is an XML language and JavaScript parser for configuring and implementing interactive audio components into web pages. It is an abstraction of the Web Audio API that uses XML syntax to describe an audio graph, and it adds features for mixing, chaining, and splitting audio signals. The syntax also offers solutions for routing and mapping variables from a hosting web page to any parameter in the audio graph. Waxml is a descriptive language and benefits from the readability of the XML format, compared with other scripting languages. The structure shares similarities with graphical environments like Max/MSP and Pure Data but is purely text-based like SuperCollider. Below is a description of the most important components and concepts of waxml. Code examples are available online.[3]

2.1 Web Audio API Abstraction

The core part of waxml is an abstraction of the Web Audio API. For almost all Web Audio node types, there is a corresponding waxml element type, like `<OscillatorNode>`, `<BiquadFilterNode>`, and `<AudioWorkletNode>`, in which the properties of the Web Audio nodes are represented by XML attributes.

To configure the audio signal routing, waxml offers a syntax to mix, chain, or split the output of any audio node. Two XML elements are specifically aimed for controlling the signal flow from its child elements: the `<Mixer>` and `<Chain>` elements. The `<Mixer>` element acts like a master bus on a mixing console and sums the output of all child elements into its output. The `<Chain>` element acts like a channel strip on a mixing console and connects the output of each child element to its next sibling where the last

[2] https://github.com/hanslindetorp/imusic.

[3] https://github.com/hanslindetorp/waxml-examples.

child element is connected to the output of the <Chain> element. This signal flow behavior could be overridden for any element by specifying its "output" attribute to any destination element(s) using a standard CSS selector syntax. If the selector results in multiple targets, the audio signal is split into multiple audio nodes.

To emulate the behavior of a "send bus" on a mixing console, there is a <Send> element with two outputs: one that acts according to the rules for any element and one that can be connected to any other element(s) using the "outputBus" attribute. The <ChannelMergeNode> and <ChannelSplitterNode> elements emulate their equivalents in WebAudioAPI using their child elements as the source or target nodes, respectively.

2.2 Custom Components

Some features require several audio nodes to build up a useful component. Examples are the <Voice> element to build polyphonic synthesizers; <Envelope> element to control audio parameters over time; <ObjectBasedAudio> element to control buffering and playback of audio files, binaural panning, and convolution reverb; and <AmbientAudio> element that controls buffering, playback of stereo or multi-channel files, and looping with crossfades to cover the loop points.

2.3 Parameter Mapping

Waxml has a <var> element for storing variable values and mapping data from the hosting web page to one or multiple audio parameters in the configuration. The element has attributes to specify expressions, including several other variables, input and output ranges, interpolation curves, value patterns, and conversion between different value domains, like MIDI note numbers to frequency. The attribute values can be a fixed value, expression containing other variables, reference to an external file containing a fixed dataset, or JavaScript function that returns a continuously updated dataset.

2.4 Event Handling

Some elements, like the <ObjectBasedAudio> or <Envelope> can be triggered by events. These events can be triggered by an event in the hosting web page or by a logical condition for variables that is met. The event is triggered once and not triggered again until the condition has become false and then evaluates true again. To receive and distribute events, waxml uses the <Event> element with the "trig" and "target" attributes where "trig" can be either a name or logical expression.

2.5 HTML Integration

Waxml is implemented into an HTML document with a script element pointing to the parser and configuration file. This file can contain any number of links to external files for separation of various parts of the configuration and reusability between different projects. Waxml offers a syntax for binding events on HTML elements to trig audio events or setting audio parameters directly in HTML without having to write any JavaScript.

Any HTML element can be set to either trig an event in waxml or update a variable according to its state. This makes it possible to connect clicks on <a>, <button>, or another element in HTML to trig an <Event> element in waxml. The connection is specified using an HTML attribute with a "data-waxml-" prefix.

2.6 External Control

In addition to controlling audio from HTML elements, waxml also has an API for Web MIDI API, [38], OSC [39], and socket.IO. It is also possible to write a custom integration with the JavaScript API in which events can be triggered and values of the <var> element can be updated to connect any interaction or data input to control waxml.

2.7 Statistical Tool

Waxml offers a simple graphical interface showing an overview of the audio configuration including a statistical tool that displays the complexity of the audio configuration. It shows all used elements with a total number for each element type, and it summarizes all mappings between external variables and audio parameters. The feature is aimed at making a fast assessment of a work.

3 METHOD

This study compares the use of waxml as a pedagogical tool from two different stages of its development. The data is collected from workshops and ongoing courses at KMH and KTH where different technologies have been used in previous years, aiming at reaching similar knowledge outcomes. The students contributed to the study by giving access to their individual project reports and source codes and by answering questions through online forms. The results from the first evaluation were reported at the Web Audio Conference 2021 [10] and were important input to the development leading up to the second evaluation. From comparisons of the results from the two stages, the impact various components had on the students' learning experiences is reported. All template files are available online.[4]

3.1 Evaluation 1

In the first stage, waxml was used in three classes at Bachelor's and Master's levels, and the students built their own interactive sound and music applications using HTML, CSS, and waxml. KTH students generally have good programming experience but less formal music training, whereas KMH students are trained in music production but lack programming experience except for some basic HTML. The three classes had independent curricula with a shared focus on sound, music, and interactivity that made them interesting for this study. The participating students spent approximately 1 week building their applications after two to three introductory lectures (around 2 h each)

[4] https://github.com/hanslindetorp/waxml-templates.

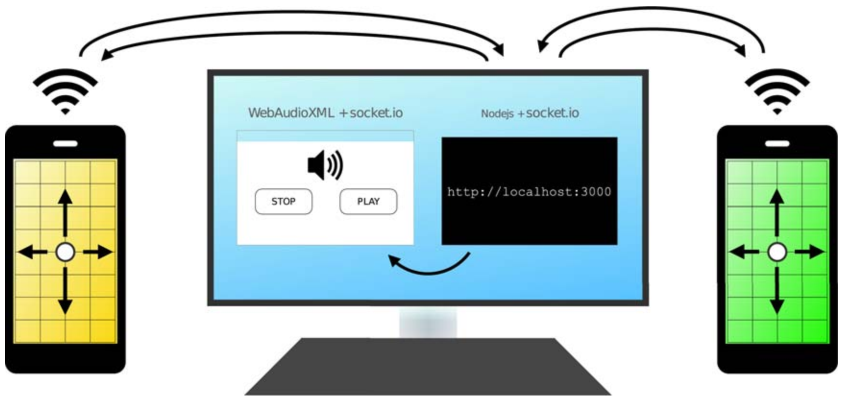

Fig. 1. The technical setup for the first evaluation at The Royal College of Music (KMH) with two smartphones connected to the web audio application using waxml, Node.js, and socket.IO.

during which the technical and aesthetic foundations were presented. All work was done from home because of the COVID-19 pandemic restrictions, and each student got a few hours of individual supervision.

The students were presented with a defined task to create interactive audio applications for smartphones. They got template files containing HTML, CSS, and JavaScript code with a set of possible connections for mapping touch events and the Device Orientation API to control audio playback. The project challenge for the students from KMH was to build a set of musical instruments for smartphones for which each device controlled a separate part of an interactive composition synchronized using iMusic, Node.js [35], and socket.IO [36] (see Fig. 1).

The students from KTH got a similar template including touch events and the task to build a sound-based, Simon Says–style game aimed at helping hard of hearing people train their listening abilities (a follow-up to earlier studies). In addition to mapping touch events to audio parameters, they also collectively wrote the necessary gaming engine, such as pattern recognition and matching.

3.2 Technical Improvements

After performing evaluation 1, the authors decided to make some changes to waxml. The most important addition in the language was an improvement of the parameter mapping described in SEC. 2.3. The authors also built an online application with a waxml code editor, waxml parser, and preconfigured variables for mapping gestures to audio parameters. This application also had an improved version of the statistical tool in which the information about the audio configuration was updated in real time.

3.3 Evaluation 2

The second part of the study was carried out 2 years after the first evaluation to test how the improvements in waxml contribute to the learning experience for the students. Three workshops were scheduled with Bachelor's and Master's students from KMH and KTH. This time, the format for the students' participation was reduced to a 2-h–long workshop focused on mapping hand gestures to sound by using an online application for coding, testing, assessment, and distribution[5] (see Fig. 2).

The application was built with the MediaPipe [11] Hands library and waxml as the main components and contained 11 presets that illustrated different mapping configurations. The workshop started with 45 min of an introduction and guided exercises, around 30 min of individual development of a demo to show for the class, 15 min of presentation, and finally around 30 min of recorded discussion. There was one active teacher who presented and assisted, and one passive teacher who observed the sessions.

3.4 Data Collection

The data from the projects and workshops were collected using three different methods: (1) source code from the students' work, (2) text and transcriptions from the students' reflections and feedback, and (3) online self-evaluation forms. The source code was analyzed with the statistical tool in waxml for counting and categorizing the number of audio elements and parameter mappings used in each project. The reflections and feedback—captured from either their written reflections (evaluation 1) or a recorded discussion after the workshop (evaluation 2)—were processed manually using content analysis with categories and quotes, focusing on how the students described the experience of using waxml and which ideas for future work the project and workshop could initiate.

In the online form for evaluation 1, the students were asked about how they spent the project time, their artistic goals, and how waxml contributed or hindered them compared with other technologies. In the form for evaluation 2, the students were asked about prior experience producing music, creating experimental music, text-based coding, programming synthesizers, and creating digital instruments. The students rated their perceived prior experience using a five-point Likert-type scale. They were also asked how they perceived the project result and were offered an optional free-text field for additional feedback.

[5] https://hanslindetorp.github.io/waxml-lab/.

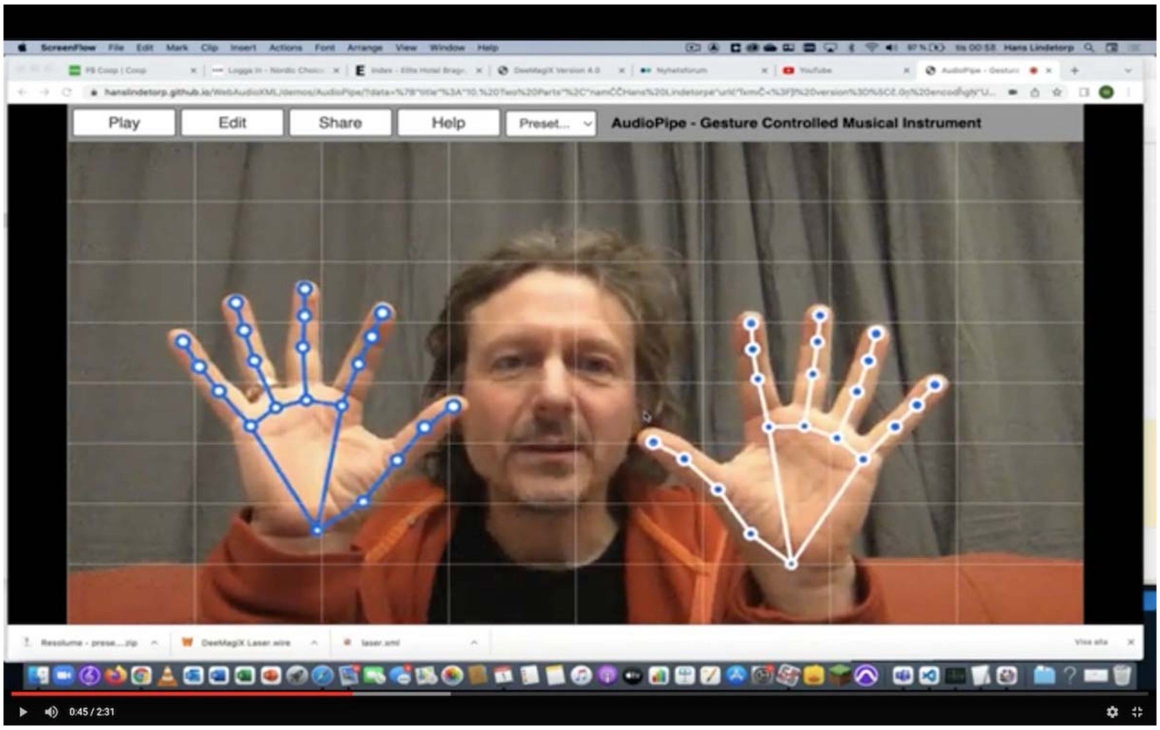

Fig. 2. The online interface used in the second evaluation showing the hand recognition feature in MediaPipe.

4 RESULTS

This section first presents a summary of the results from the evaluation round that was reported at the Web Audio Conference 2021 [10]. The results from the second evaluation round are presented chronologically according to the process: prior experience, source code analysis, self-assessment of the result in relation to their intentions, and feedback from the reflective discussion with thematically ordered comments, quotes, and patterns.

4.1 Evaluation 1

The first evaluation had a series of three project work periods: one at KTH with eight Master's-level students (four female and four male) and two at KMH with 12 Bachelor's-level (12 male) and seven Master's-level (one female and six male) students. They worked from home and received a few hours of supervision through video-conferencing software. All managed to reach their goals during the project week. A summary of the most important findings follows below.

The three groups worked with different template files and challenges for their projects. The complexity varied a lot between the different projects, and there were both simple and complex configurations represented in each group. Generally, Master's students at KMH had more complex projects than Bachelor's students. The KTH students got a limited template file and tended to build their audio configuration from the bottom up, whereas the KMH students would pick and choose from a more complex template and replace temporary audio files with new ones.

The feedback received from the students' reflective texts and online form indicated that the technology came with both drivers and barriers for the students' creative flow. One student reported to have "a bit of a hard time understanding how everything was working" (P1), and another said, "when I learned all the parameters in waxml the process went on being much easier than I thought it would be" (P2). Students with little prior programming experience complained that small errors in the code could (momentarily) break the whole application, but there was also encouraging feedback like "what I could learn during this time opened for a new way of thinking and to 'step outside the box' [musically]" (P3).

4.2 Observations From First Evaluation

When summarizing the first evaluation, it was concluded that the technology worked surprisingly well on all different platforms including Apple and Windows computers and iOS and Android smartphones. Waxml also proved itself beneficial as a coding platform for the students, removing the need to learn JavaScript first. The KTH projects aimed to help hard of hearing people, and it benefited from the affordances offered by web technologies for which distribution can be as simple as sharing a link. A feature that was developed for the first evaluation was the statistics tool in waxml that made it possible to analyze the audio configuration and assess the work more easily.

The insights from the first evaluation led to further development of waxml in which barriers were addressed and new features were added. Several online applications were

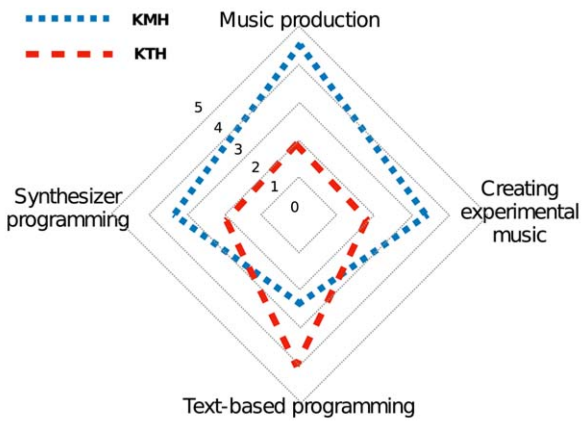

Fig. 3. Average values (1–5) of the students' prior experience grouped by campus. Dotted lines are The Royal College of Music (KMH) and dashed are KTH Royal Institute of Technology (KTH).

also developed on top of waxml, from which solutions for a better coding environment have been explored.

4.3 Evaluation 2

The second evaluation round had three workshops: two at KMH with two Bachelor's-level (one female and one male) and eight Master's-level (eight male) students and one at KTH with five Master's-level students (one female and four male). They worked independently with a minimal supervision for 30 min and created one unique gesture-controlled audio application each. All students managed to map at least one gesture to an audio parameter during the session.

The participants self-evaluated their prior experience of music production, creation of experimental music, synthesizer programming, and text-based programming using a five-step Likert-type scale. The result shows that the students from KMH are rating their experience of music-related activities higher than the students from KTH, who, on the other hand, rated their experience of text-based programming higher than the students from KMH. This is illustrated in Fig. 3.

The analysis of the source code using the statistical tool indicated that all students began their work with one of the available presets and changed them to various degrees. That typically involved changing settings for the parameter mapping and/or adding new nodes to the current configuration. More than two thirds of the students had changed or added 40% or more of the audio nodes. Two students did not manage to deliver their work at all, and the rest changed only a single value. When self-evaluating to what degree they managed to achieve their goals, six answered that they succeeded almost or completely, one thought to not have succeeded at all, and the remaining eight students perceived their result to be somewhere in between.

The output from the group discussions contains reflections relating to the experience, engaging factors, new ideas, and suggestions for improvements on the application. The students generally described the experience as overwhelmingly positive with expressions like "nice," "super fun," "super flexible," "really simple," and "super cool."

Several answers point out that it was easy to understand and map ideas, and it offered a direct way of exploring the relations between gestures and sounds. They value that it was fast to get started, that they could make something creative after only 30 min, and that the application had no dependencies on any external gadgets. Most students agreed that experimenting to find out an intuitive way of using gestures to make different sounds was among the most engaging factors of the workshop.

Critical feedback referred only to coding, for which one student mentioned that the workflow to copy code snippets and change specific values felt non-intuitive and two students pointed out that it takes some time to learn the variable naming structure. One student confessed to be a bit frightened to write code because "you write one wrong character and then nothing works" but also said that waxml "was more forgiving compared to languages like C#" (P4).

4.4 Observations From Second Evaluation

One part of the reflective discussion covered new ideas and what potential the students could see after they had participated in the workshop. Four students focused on the artistic aspects of using the body to control sound and presented ideas like choreographing a dance for the hands playing together with an acoustic instrument or combining dancing with sound generation. Others pointed out the inclusive aspect of using the body as a controller and imagined a face-controlled interface where even people with severe physical disability would be able to make music. One student saw the potential for people with movement restrictions in their hands to be encouraged to exercise in order to make sound with the gestures.

The KMH Master's students mainly contributed with ideas for improving their artistic practice, imagining gesture-controlled features for working with music production. One student envisioned a more human use of technology in which dynamics and variations in the mix could be controlled with body movements rather than by drawing automation curves in a Digital Audio Workstation or setting parameters with numeric values.

The students suggested some improvements to the application to make it more creative and easier to use. Several students wished that there was a built-in documentation of the waxml language in which all possible nodes were presented and explained. Others pointed out that debuggers would be helpful, both for the XML syntax and audio configuration. Some suggestions concerned the interface and proposed a graphical interface with visual blocks and drag-and-drop features. They also asked for a combined view in which the code, the video, a dynamic grid, and a visual inspector could be visible at the same time. One student mentioned a future scenario in which the gestures themselves could act as a way of coding the relation between them and the sound.

Finally, the discussion opened for more general questions about limitations, creativity, and technology. One student argued that the technical limitations in a system are not barriers but rather driving factors for creativity in artistic work—similarly to how traditional musical instruments encourage creativity because of their limitations. The discussion also covered the benefits of using web technologies for making music, and it was pointed out that a unique opportunity with web technology, and in particular JavaScript, is that it is extremely well-supported and that it is easy to take sensors, gadgets, and APIs intended for other purposes and convert them into musical instruments.

4.5 Limitations

The main limitation of the study is the lack of generalizable results. Three distinctly different student groups with both Bachelor's and Master's students were engaged. Although the KMH curriculum involves music, the KTH curriculum is primarily centered around engineering and programming; however, the students at both universities all demonstrated skills in both domains and thus showed a greater homogeneity than diversity. There is an over-representation of male students, particularly at KMH.

5 DISCUSSION

This section compares the result from the two rounds of evaluation and discusses what changes in the preparations might have had the most impact. These findings are also compared with related studies.

5.1 Efficiency and Focus

When comparing evaluations 1 and 2, the preparation time was reduced from two to three lectures plus time for installation of coding software and template files down to 45 min of introduction and exercises for the recent workshops. This seems to have been enough for the students getting started, and there were no compatibility issues on any of the students' computers. This is a very promising result for web audio and contributes to the authors' efforts of getting rid of irrelevant technical obstacles to stay focused on the main learning objectives. It is also worth noticing that there was hardly any need for assistance during the recent workshop and that the integrated interface for development, documentation, and testing reduced the start-up time and encouraged a more iterative process of moving between coding and testing.

5.2 User Experience

Both evaluations received quite positive feedback when the participants described their experiences of working with waxml, but it was obvious that the MediaPipe library added a very attractive layer to the workshop. Many of the participants referred to the connection between gestures and sound as the most engaging factor, and it was striking how they got drawn in to trying out the interactions as soon as they tried a configuration. Some of the students were more engaged in playing the instrument than building or optimizing it.

An interesting detail observed during the workshop was that the students' playing styles were radically different; some used piano-like finger movements, some made large hand gestures, and others again appeared to paint in the air. It is worth noticing that the only new feature in the waxml that was useful for this evaluation was the `<var>` element. It added a lot of mapping features that made complex mapping easier than before, but because the students did not know any other method, they took it for granted. The fact that they used it without complaining or failing is probably a sign of the feature being well designed.

In the first evaluation, different behaviors that might have been caused by different template files were observed. In the most recent workshop, there were 11 presets with different audio configurations available from a drop-down menu. This seems to have been a step toward a setup that encourages the student to be freer with their creativity but still have some models to start from. A potential for a future version could be to prepare the tool with smaller blocks of elements that could easily be put together.

5.3 Comparing Learning Outcomes

The evaluations assessed two quite different activities and were therefore not expected to leave the students with the same learning outcome, but it is possible to compare the depth of reflections and number of new ideas for creativity they caused. Interestingly, the recent 2-h workshop in the second evaluation seems to have sparked at least as many new ideas and perspectives as the previous course spanning over a 2-week period. The efficiency is likely one reason for this, but the authors also think that the gesture recognition feature itself played a big role. Many of the reflections from the second evaluation referred to the ability to control sound with the body and how the body is attractive to work with instead of a keyboard and mouse.

The statistical feature was an important result of the first evaluation, and it was further developed to integrate into the tool used in the second evaluation. It further reduced the time spent on assessment, but after comparing the data from all projects, it is still not clear what the data actually tells about the students' accomplishments. The authors argue that the feature could work for clearly directed tasks and that it would be optimal to make it compatible with learning management systems. For assessment of the artistic contents, it is necessary to further investigate whether teachers can be helped by using the statistical analysis tool.

5.4 A Comparison With Related Studies

The authors contribute to studies on similar frameworks by collecting data from a relatively large group of artistic and creative students. Similarly to BRAID [19], waxml aims at reducing barriers to web audio instrument design, and it is also concluded that the sharing aspect of such a framework is important. The authors of the study on BRAID argued that a collaborative interface could open great possibilities for education and ensemble work, which the authors can confirm from their experiences. Compared to the study on EarSketch [20], the current authors' aims differ in several ways. EarSketch seeks to engage students in computing through music, whereas waxml aims at supporting students in producing interactive sound and music with as little programming as possible. Still, some common goals were reached, with students from both projects becoming more interested in sound and music computing.

5.5 Accessibility and Distribution

The first evaluation was an important indicator for which it could be seen that Web Audio contributed greatly to make the courses accessible, even through the COVID-19 lockdown. The second evaluation looked at the accessibility aspect one step further, in which both the application and final project could be distributed in a standard URL. This feature encouraged the students to share their work through text messages and social media and try each other's instruments continuously. For the future, it would be interesting to see how a more-developed online coding framework could benefit the learning experience, allowing for collaboration on the same project.

6 CONCLUSION

Web Audio was evaluated for learning, and it is concluded that the technology is very useful and easy to distribute. Waxml proved to make audio configuration accessible for creative artists, and the online tool for developing gesture-controlled audio applications proved to be very efficient, robust, and fun to use. It made it possible to introduce students to concepts like sound synthesis instrument design, interactive audio, parameter mapping, and coding in a 2-h workshop, and it removed most of the obstacles found in earlier classes and reduced the preparation time before the students could work independently to less than 1 h.

The evaluations were done with three groups of engineering and music production students at different levels and universities; it is unlikely that the authors' findings can be generalized. However, the long list of previous investigations during the past several years and the results here indicate that there is a great potential, in general, for creative web audio tools such as waxml. The authors also argue that even if the audio implementation could have been done using other abstractions and approaches, a standardized format like waxml could be beneficial for sharing components between different web audio applications.

7 ACKNOWLEDGMENT

The authors thank all participating students at KMH and KTH. They are also grateful to the organizers and jury members of the Web Audio Conference 2021 for the Best Paper Award.

8 REFERENCES

[1] P. Adenot and H. Choi (Eds.), "Web Audio API," W3C Recommendation (2021 Jun.). https://www.w3.org/TR/webaudio/.

[2] T. Schaller and I. G. Burleigh, "Using Web Audio API in Web-Based Tools for Ear Training of Sound Engineers," in *Proceedings of the AES UK 26th Conference: Audio Education* (2015 Aug.), paper 17.

[3] J. Freeman, B. Magerko, D. Edwards, et al., "EarSketch: Engaging Broad Populations in Computing Through Music," *Commun. ACM*, vol. 62, no. 9, pp. 78–85 (2019 Sep.). http://doi.org/10.1145/3333613.

[4] W. Turner and S. Leonard, *JavaScript for Sound Artists: Learn to Code With the Web Audio API* (Routledge, New York, NY, 2017). http://doi.org/10.1201/9781315659732.

[5] A. Xambó, R. Støckert, A. R. Jensenius, and S. Saue, "Facilitating Team-Based Programming Learning With Web Audio," in *Proceedings of the International Web Audio Conference*, pp. 2–7 (Trondheim, Norway) (2019 Dec.).

[6] A. Xambó, R. Støckert, A. R. Jensenius, and S. Saue, "Learning to Code Through Web Audio: A Team-Based Learning Approach," *J. Audio Eng. Soc.*, vol. 68, no. 10, pp. 727–737 (2020 Oct.). http://doi.org/10.17743/jaes.2020.0019.

[7] H. Lindetorp, "Immersive and Interactive Music for Everyone," in *Proceedings of the Nordic Sound and Music Computing Conference (SMC)*, pp. 16–20 (Stockholm, Sweden) (2019 Nov.).

[8] H. Lindetorp and K. Falkenberg, "WebAudioXML: Proposing a New Standard for Structuring Web Audio," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 25–31 (Torino, Italy) (2020 Jun.). http://doi.org/10.5281/zenodo.3898655.

[9] H. Lindetorp and K. Falkenberg, "Audio Parameter Mapping Made Explicit Using WebAudioXML," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 352–358 (Online) (2021 Jun.). http://doi.org/10.5281/zenodo.5038686.

[10] H. Lindetorp and K. Falkenberg, "Putting Web Audio API to the Test: Introducing WebAudioXML as a Pedagogical Platform," in *Proceedings of the International Web Audio Conference* (Barcelona, Spain) (2021 Jul.).

[11] C. Lugaresi, J. Tang, H. Nash, et al., "MediaPipe: A Framework for Perceiving and Processing Reality," in *Proceedings of the 3rd Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR)* (Long Beach, CA) (2019 Jun.).

[12] A. Walker, H. Leary, C. Hmelo-Silver, and P. A. Ertmer (Eds.), *Essential Readings in Problem-Based Learning: Exploring and Extending the Legacy of Howard S. Barrows* (Purdue University Press, West Mafayette, IN, 2015).

[13] T. Berners-Lee, "W3C Mission," https://www.w3.org/Consortium/mission (accessed March 14, 2022).

[14] H. Choi and J. Berger, "WAAX: Web Audio API eXtension," presented at the *Proceedings of the Inter-*

national Conference on New Interfaces for Musical Expression, pp. 499–502 (Daejeon, Korea) (2013 May). http://doi.org/10.5281/zenodo.1178494.

[15] Y. Mann, "Interactive Music With Tone.js," in *Proceedings of the International Web Audio Conference* (Paris, France) (2015 Jan.).

[16] S. Ren, L. Pottier, and M. Buffa, "Build WebAudio and JavaScript Web Applications Using JSPatcher: A Web-Based Visual Programming Editor," in *Proceedings of the International Web Audio Conference* (Barcelona, Spain) (2021 Jul.).

[17] A. Thompson and G. Fazekas, "A Model-View-Update Framework for Interactive Web Audio Applications," in *Proceedings of the 14th International Audio Mostly Conference: A Journey in Sound*, pp. 219–222 (Nottingham, UK) (2019 Sep.). http://doi.org/10.1145/3356590.3356623.

[18] I. G. Burleigh and T. Schaller, "Quint.js: A JavaScript Library for Teaching Music Technology to Fine Arts Students," in *Proceedings of the International Web Audio Conference* (Paris, France) (2015 Jan.).

[19] B. Taylor and J. Allison, "BRAID: A Web Audio Instrument Builder With Embedded Code Blocks," in *Proceedings of the International Web Audio Conference* (Paris, France) (2015 Jan.).

[20] A. Mahadevan, J. Freeman, and B. Magerko, "An Interactive, Graphical Coding Environment for EarSketch Online Using Blockly and Web Audio API," in *Proceedings of the International Web Audio Conference* (Atlanta, GA) (2016 Apr.).

[21] E. Lakka, M. Papadaki, D. Brutzman, R. Puk, and A. G. Malamos, "X3D Audio Graph for the Consistent Declarative Representation of the W3C Audio API," in *Proceedings of the 26th International Conference on 3D Web Technology*, paper 8 (Pisa, Italy) (2021 Nov.). http://doi.org/10.1145/3485444.3487645.

[22] L. Wyse and S. Subramanian, "The Viability of the Web Browser as a Computer Music Platform," *Comput. Music J.*, vol. 37, no. 4, pp. 10–23 (2013 Dec.). http://doi.org/10.1162/comj_a_00213.

[23] C. B. D. Clark and A. Tindale, "Flocking: A Framework for Declarative Music-Making on the Web," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 1550–1557 (Athens, Greece) (2014 Sep.).

[24] C. McCormick and S. Piquemal, "WebPd," github.com/sebpiq/WebPd (accessed February 18, 2020).

[25] C. Roberts, G. Wakefield, M. Wright, and J. Kuchera-Morin, "Designing Musical Instruments for the Browser," *Comput. Music J.*, vol. 39, no. 1, pp. 27–40 (2015 Mar.). http://doi.org/10.1162/comj_a_00283.

[26] V. Lazzarini, E. Costello, S. Yi, and J. Fitch, "Csound on the Web," in *Proceedings of the Linux Audio Conference*, pp. 77–84 (Karlsruhe, Germany) (2014 May).

[27] E. Hong and J. Kim, "Interactive Web Audio Networking System and Method With DrSax.js," in *Proceedings of the 3rd International Conference on Communication and Information Processing*, pp. 411–414 (Tokyo, Japan) (2017 Nov.). http://doi.org/10.1145/3162957.3162968.

[28] S. Robaszkiewicz and N. Schnell, "Soundworks – A Playground for Artists and Developers to Create Collaborative Mobile Web Performances," in *Proceedings of the International Web Audio Conference* (Paris, France) (2015 Jan.).

[29] C. Baumann, J. F. Baarlink, and J.-T. Milde, "Body Movement Sonification Using the Web Audio API," in *Proceedings of the International Web Audio Conference* (Berlin, Germany) (2018 Sep.).

[30] J.-O. Gullö, I. Höglund, J. Jonas, et al., "Nobel Creations: Producing Infinite Music for an Exhibition," *Dansk Musikforskning Online*, pp. 63–80 (2015 Special Edition).

[31] R. Bresin, L. Elblaus, E. Frid, et al., "Sound Forest/Ljudskogen: A Large-Scale String-Based Interactive Musical Instrument," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 79–84 (Hamburg, Germany) (2016 Aug.). http://doi.org/10.5281/zenodo.851191.

[32] J. Paloranta, A. Lundström, L. Elblaus, R. Bresin, and E. Frid, "Interaction With a Large Sized Augmented String Instrument Intended for a Public Setting," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 388–395 (Hamburg, Germany) (2016 Aug.). http://doi.org/10.5281/zenodo.851289.

[33] E. Frid, H. Lindetorp, K. F. Hansen, L. Elblaus, and R. Bresin, "Sound Forest: Evaluation of an Accessible Multisensory Music Installation," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, paper 677 (Glasgow, Scotland) (2019 May). http://doi.org/10.1145/3290605.3300907.

[34] H. Frisk and W. Brunson, "Building for the Future - Research and Innovation in KMH's New Facilities," in *Proceedings of the Sound and Music Computing Sweden: Bridging Science, Art, and Industry*, pp. 10–11 (Stockholm, Sweden) (2014 Dec.).

[35] OpenJS Foundation, "Node.js Is a JavaScript Runtime Built on Chrome's V8 JavaScript Engine," https://nodejs.org/ (accessed March 18, 2022).

[36] D. Arrachequesne, "Socket.IO: Bidirectional and Low-Latency Communication for Every Platform," https://socket.io/ (accessed March 18, 2022).

[37] H. Lindetorp and K. Falkenberg, "Sonification for Everyone Everywhere – Evaluating the WebAudioXML Sonification Toolkit for Browsers," in *Proceedings of the International Conference on Auditory Display*, pp. 15–21 (Online) (2021 Jun.).

[38] MDN, "Web MIDI API," https://developer.mozilla.org/en-US/docs/Web/API/Web_MIDI_API (accessed May 21, 2022).

[39] M. Wright and A. Freed, "Open SoundControl: A New Protocol for Communicating With Sound Synthesizers," in *Proceedings of the International Computer Music Conference*, vol. 1997 (Thessaloniki, Greece) (1997 Sep.).

THE AUTHORS

Hans Lindetorp

Kjetil Falkenberg

Hans Lindetorp studied music pedagogy at KMH Royal College of Music in Stockholm where he teaches music production at the Music and Media Department. Hans is also a doctoral student in the Division of Media Technology and Interaction Design, KTH Royal Institute of Technology, with a thesis on music production for interactive media. His key interests are technology in the service of humans and music, web audio, and design for all.

•

Kjetil Falkenberg studied pedagogy and musicology at NTNU Norwegian University of Science and Technology, Trondheim. He became a Doctor of Technology in Speech and Music Communication at the Division of Media Technology and Interaction Design, KTH Royal Institute of Technology, with a thesis on modeling DJ scratching. He is a Docent in Media Technology from Södertörn University and currently associate professor in sound and music computing at KTH. His key research interests are sound and music in interaction, sound design, sonification, accessibility, and design for all.