Quint.js: A JavaScript library for teaching music technology to fine arts students

Ian George Burleigh
Dept. of Music, Digital Audio Arts
University of Lethbridge
4401 University Drive
Lethbridge, AB T1L 3M4, Canada
ian.george.burleigh@gmail.com

Thilo Schaller
Dept. of Music, Digital Audio Arts
University of Lethbridge
4401 University Drive
Lethbridge, AB T1L 3M4, Canada
thilo.schaller@uleth.ca

ABSTRACT

This paper presents `quint.js`, a JavaScript library for making interactive HTML/SVG/Web Audio "applets". Two-dimensional geometric structures ("machines") that are based in SVG vector graphics with audible feedback synthesized through Web Audio are used to demonstrate various physical, acoustic, and psychoacoustic phenomena and are applied in teaching music technology courses to fine arts students. The current core `quint.js` library and extension modules are a starting point for more extensive integration of Web Audio to support the delivery of course content and for the creation of integrated, interactive lecture notes.

Categories and Subject Descriptors

[**Applied computing**]: Computer-assisted instruction; Sound and music computing; Interactive learning environments.

General Terms

Design, Human Factors.

Keywords

Audio arts, audio engineering, instruction, ear training, Web Audio, SVG, JavaScript.

1. INTRODUCTION

Technological changes in storage, processing, and delivery of information are forcing fundamental changes in education, especially in higher education. Printed books and physical audio-visual media are being replaced by "digital libraries" with learning materials that are delivered to computing devices — notebooks and tablets. In a "hybrid classroom", the traditional teacher-student relationship is changing into "hybrid pedagogy", a concept that can be described as "hybrid learning" happening in the classroom and online, in "an intersection of physical and virtual learning space" [2].

HTML5 and related web technologies support the development of cross-platform multimedia applications that run in standard-compliant web browsers. This opens a possibility for creating interactive materials that combine both linear and non-linear[1] learning, that can be accessed anywhere on almost any computing device without the need to install additional software.

Digital Audio Arts at the University of Lethbridge is a Bachelor of Music program that prepares students for work in "sonic arts" — audio engineering and production, computer-assisted music, electroacoustic music composition, and related areas of music technology [1]. The students work within a fine arts environment and as such have affinity for arts rather than for sciences and engineering. However, depending on their field of specialization, they still have to comprehend fundamental concepts of Euclidean geometry, Newtonian physics, acoustics, psychoacoustics, and 'harmonics' (i.e., the science of musical sound, specifically issues of tuning, timbre, and temperament [3, 15]). The students work with microtonality in computer-assisted music composition and performance — a modern sonic artist is "not a musician, but a worker in frequencies and intensities" [17]. They also have to acquire vocational skills needed for audio engineering such as the ability to identify bands of the audible frequency spectrum, critically assess sonic results of signal processing, or to recognize flaws in recorded or synthesized sound.

In this learning environment, teaching of scientific concepts and training of vocational skills are most effective through intuitive auditory and/or visual demonstration and experimentation. The audio synthesis and processing capacity of Web Audio in combination with SVG graphics is a platform for development of interactive, animated "applets"[2] with audible feedback that serve as efficient learning tools.

2. QUINT.JS

`quint.js` (Quint[3]) is a medium-sized JavaScript library. In its "minified" form, Quint is about 30kB in size.

---

[1] *Linear learning* follows the classic "textbook", step-by-step sequence. In *non-linear* learning there are multiple possible paths through hyper-linked materials.

[2] The term 'applet' is used here simply as "a small application (program) placed on a web page"; by itself it is not meant to imply any particular software technology or architecture.

[3] Shortened from *quinta*, a name for the perfect fifth (musical interval).

The purpose of Quint is to support and simplify the creation of two-dimensional interactive geometric structures in SVG (within the HTML `<svg>` tag), such as one would make in the past, for example, on paper using a compass and a ruler, or in a graphing calculator. To synthesize sound, Quint originally required a local running instance of Csound [18] that was controlled through Web Sockets connection and a custom-made minimal Web Sockets server. The introduction of Web Audio was the "last piece of the puzzle" that was needed to run everything in a browser; now only Quint and supporting JavaScript modules (Quint relies on `d3.js` [4] to manipulate DOM elements) are required to deliver all audio-visual content.

Quint is a free software, released under the MIT licence on GitHub.[4] Tutorials, documentation, and examples are available on the Quint website [5].

A Quint applet is embedded into an HTML page through a named `<div>` or `<span>` element. Everything else — the SVG container, SVG groups and elements, as well as other HTML elements — are created by manipulating the DOM from JavaScript code.

Quint consists of several modules. The core module (`quint.js`) contains:

- Code that sets up the main applet container, its geometry, visual appearance (border, title, etc.), and a container for optional user interface elements (HTML `<input>`).
- Code for manipulation of SVG elements and their attributes, SVG groups and nested groups.
- Handlers of interactive interface events; in particular, mouse events.
- Code that supports 2D vector algebra operations; work with Cartesian and polar coordinate systems, local and global coordinates, affine transformations, etc.
- Various helper functions for easier programming, such as support for classical class inheritance, functional-style iterators, and debugging functions.

The interactivity of Quint applets is based on a simple concept: selected SVG elements can be made "movable". Movable elements respond to `mousedown` (click) and `mousemove` (drag) events. A handler function associated with a movable element can, in response, programmatically move or otherwise manipulate other elements; the positions of each movable element can be constrained to a particular area, set of positions, a path, etc.

The core Quint module supports handling of the SVG elements `ellipse`, `circle`, `line`, `rect`, and `path`, operations that set or retrieve their geometric and visual attributes (position, radius, size, stroke and fill colours, width, etc.) and operations of vector algebra specifically related to the geometry of the respective elements. The `path` element is used to construct spline curves through a sequence of points (Cartesian coordinates) that can be scaled and translated within the coordinate space. (The typical use of spline curves is to display wave shapes.) SVG `text` elements can be added as labels or annotations. Finally, the SVG `foreignObject` element can be used to include HTML elements, namely HTML canvas.[5]

The core module also provides a set of frequently used abstractions, for example the construction of grid lines, or user interface elements implemented in SVG such as movable "handles" for user interaction, sliders, or checkboxes.

The Quint user interface module (`quint_ui.js`) supports HTML (`<input>`) elements: `button`, `radio`, `checkbox`, `number` and `range`, and their interaction with the SVG elements. The audio (`quint_audio.js`) module contains various wrapper and helper functions for Web Audio.

Quint has several extension modules that contain code that had been first developed for a specific purpose and then abstracted as a generic solution:

- Waveforms (`quint_waves.js`): code that constructs spline curves in the shape of sinusoidal waves specified by their frequency, amplitude, and phase and complex waves synthesized as a superposition of component (partial) sinusoidal waves.
- Max/Pd-like [13] data-flow "patcher" (`quint_patcher.js`): an abstraction of movable boxes (nodes) with inlets, outlets, and user interface that can be connected by audio signal and events-carrying "patch cords" to create a data-flow network.
- A set of "unit generators" (`quint_ugens.js`) for use in a patcher: various types of oscillators, number controls, sliders, gain controls, oscilloscope, etc.

3. EXAMPLE APPLETS

The following paragraphs describe selected applets that were created for use in Digital Audio Arts courses:

3.1 Circular Motion

Fig.1 shows an applet that is used to demonstrate several fundamental principles of trigonometry and wave motion: the relations between circular motion and simple harmonic motion, angular and temporal frequency, right-angled triangle inscribed into a unit circle, sine and cosine values, etc. All these concepts are essential for the delivery of acoustics, sound synthesis, and audio production courses. The yellow handle (a small disc) can be moved around the perimeter of the unit circle on the left and the sine and cosine waveforms are updated accordingly.

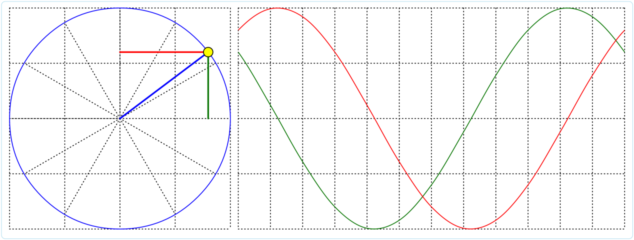

Figure 1: Circular Motion applet

---

[4] `https://github.com/quinta-audio/quint.js`

[5] As of writing this text, using `foreignObject` is slightly problematic. See `https://code.google.com/p/chromium/issues/detail?id=371724`.

3.2 Additive Synthesis

The applet in Fig.2 demonstrates how complex, band-limited waveforms are synthesized from a series of partials (partial simple waveforms). The user can adjust the amplitude and phase of each partial by dragging the corresponding handles. The constructed complex wave is displayed in the top part of the applet; the audible result is synthesized by a bank of Web Audio oscillators and delay lines.

The user can choose from several waveform presets (sine, triangle, rectangular, and sawtooth wave) or create an arbitrary waveform by changing the amplitudes and phases of each partial. The combination of visual and auditory presentation offers an intuitive approach to associate wave shapes with timbre ("sound colour") and helps students to recall the relative amplitudes and phases of the partials that constitute common waveforms.

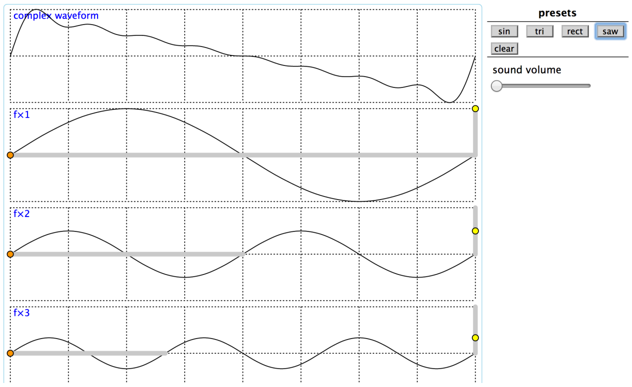

Figure 2: Additive Synthesis applet

3.3 Fourier

This applet is the first one in a planned series of applets to demonstrate the principles of Fourier analysis in terms that are accessible to music students. The applet is complementary to the Additive Synthesis applet and offers the following intuitive explanation of continuous Fourier transform:[6] A complex (periodic) waveform has been synthesized by adding a number of simple sinusoidal waves. The task is to isolate the component waveforms and thus break the complex waveform into the original set of sinusoidal waves ("partials"). The complex waveform (its selected part) is multiplied by a "probe" sinusoidal wave and the product is "integrated" (summed). Finding a positive local maximum by varying the probe wave frequency and phase means that a component of the complex waveform has been located.

Fig.3 shows the relevant part of the applet: the synthesized waveform, the probe sinusoidal wave — its amplitude, frequency, and phase — and the wave product with an orange-coloured indicator that shows its positive magnitude.

Synthesis by a bank of oscillators and delay lines (to control the phase of component waves) through Web Audio provides an auditory feedback. As the probe wave begins to match a component wave, auditory "beats" can be heard. Eliminating the beats, in a process similar to tuning of musical instruments, corresponds with isolating the component wave. This exercise naturally contributes to the development of listening skills while learning a theoretical concept.

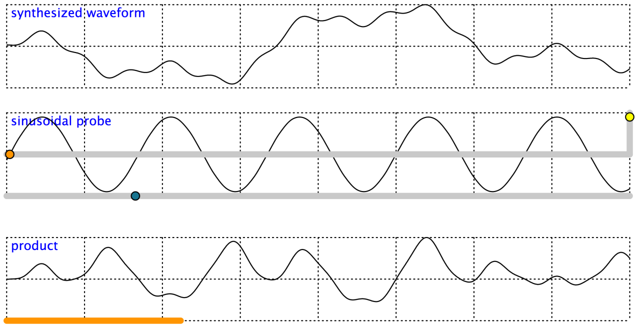

Figure 3: Fourier applet

3.4 Pitch Spiral

The "pitch spiral" applet, shown in Fig.4, is used not only to demonstrate but mainly to experiment with aspects of the psychoacoustical theory of *sensations of tone*, a major work of H. Helmholtz [8]. The theory can be summarized as follows: Complex musical tones are formed from a series of simple tones. The interference of two simple tones is perceived as beats (variations of the aggregate amplitude) if the difference of frequencies is below a threshold of approx. 16 Hz, as an unpleasant *sensory roughness* if the difference is around 30 Hz, and as a *difference tone* if the frequency difference is higher than that. Tuning two or more complex tones into a pleasant sonority requires that the aggregate sensory roughness among pairs of their constituting simple tones is minimized.

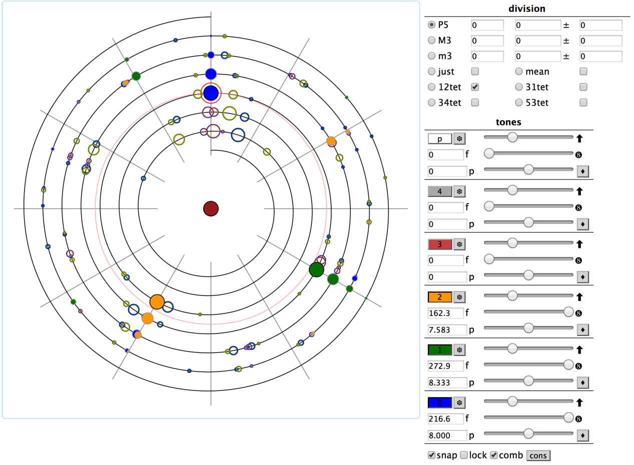

Figure 4: Pitch Spiral applet

The spiral represents a flattened helical model of musical pitch. Each turn of the spiral (in the inside-out direction) corresponds to raising the pitch by one octave. Various polar grids can be overlaid on the spiral, indicating: a) the pitches of twelve (shown in the figure), 31, 34, or 53-tone equal division of the octave, b) pitches of various tuning systems (Pythagorean, meantone, just), and c) chains of fifths or thirds (the two musical intervals essential to Western harmony) that can also be tempered at will. Coloured

---

[6] $S(f) = \int_{-\infty}^{\infty} s(t) \cdot e^{-i\omega t}\,\mathrm{d}t$

discs represent the partials of several test complex tones; the hollow small circles represent the various difference tones. Fig.4 shows the partials and generated difference tones of a major triad in 12-tone equal division of the octave; their non-coincidence explains why an equally-tempered triad is “strangely uneasy, and no wonder” [12].

The visual representation of the spiral pitch space with complex tones and their constituting partials and difference tones related to various theoretically-derived grids, together with corresponding synthesized sound allows many experiments by Helmholtz to be re-created in the browser.

3.5 Noise Spiral

The “noise spiral” (Fig.5) is a tool for one of the most elementary stages of technical ear training: frequency band recognition.

Especially for students with a focus on audio engineering, ongoing technical ear training is an essential activity. Listening skills take time to develop; the Noise Spiral applet is designed to give students the opportunity for regular practice.

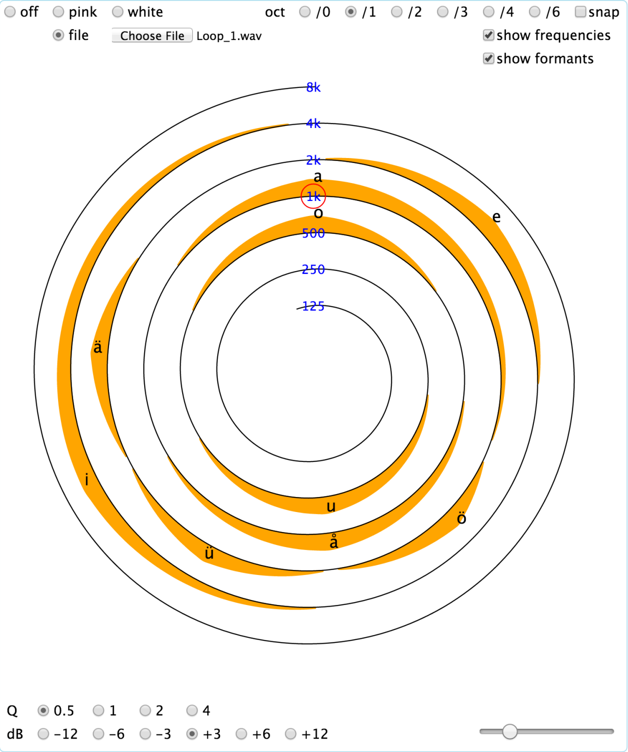

Figure 5: Noise Spiral applet

For sound engineers, composers, and performers of electronic music, recognizing the sonic characteristics of a specific frequency band is an important skill whether working in the recording studio or in a live sound environment. In both environments, decisions on equalizer settings are made after an aural assessment of the sound material and the identification of excess or deficiency of energy within certain frequency bands. In order to find the desired band, students are often taught to raise the gain of an equalizer and swipe through the frequency spectrum. However, the imagined sonic goal might change or be lost due to the exposure to various alterations of the signal while searching for the desired part of the spectrum [16]. Through experience and technical ear training, the recognition of frequency bands can be accelerated and ideally the need for swipes eliminated [6]. Approaches to developing this skill often use filtered pink noise in addition to musical material [7, 9, 11], to have a constant presence of all frequency bands, which makes any changes to the spectrum consistent and immediately audible.

The Noise Spiral has been created as a practice tool for frequency band recognition to accelerate the development of listening skills. The applet uses pink noise that is filtered by a “peaking” Web Audio biquad filter. Students are asked to identify the centre frequency of boosts or cuts. The user can select the Q and gain of the filter and the type of the source signal. Selecting the division of the octave (1/2, 1/3, . . .) changes the number of frequency bands and decreases the distance between adjacent steps, thus making the exercise gradually more difficult.

The visualization of the frequency spectrum is based on the same graphical model that is used for the Pitch Spiral. The spiral optionally displays main formant areas of vowels. The relation to human speech appears to serve as a valuable addition within ear training sessions and for many beginning students, this additional visual and experience-based reference acts as a shortcut to master the recognition of frequency bands. While the system is currently based on German vowels [10], other language overlays — such as English, Finnish, male/female [14] — are considered for future updates.

In order to put the listening skills into a realistic context, users can also import audio files for processing. This way, one can decide to either work with individually recorded instruments or a finished stereo mix and thus create a realistic situation for application of the listening skills. The goal of this feature is to enable students to identify frequency bands within the spectrum of various sound sources in order to address problems or, use sonic characteristics of specific frequency bands of instruments for artistic purposes.

3.6 Patcher

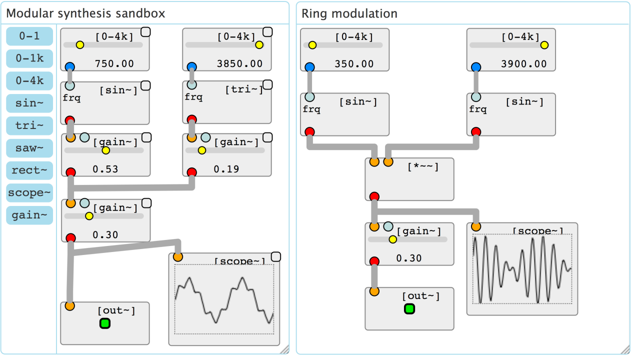

Figure 6: Patcher

The Patcher is a proof-of-concept realization of a modular audio synthesis toolkit. Its concept is based on the well-known Max/Pd-like signal flow abstraction: graphical boxes represent signal and data-processing “objects” with “inlets” and “outlets” connected by “patch cords” [13].

The Patcher is being developed as an “extension” of the `quint.js` library, using a number of its features. It drives further development of Quint: for example, the support for HTML canvas through SVG `foreignObject` was required to implement the "oscilloscope" object.

The main, immediate purpose for Patcher is to illustrate the fundamental principles of various audio synthesis methods, such as ring modulation (Fig.6).

4. CONCLUSIONS AND FUTURE WORK

The web browser platform and web technologies are ready for integration of text, animated visual content and real-time synthesized audio without relying on traditional technologies external to the browser, such as Flash or Java.

We developed `quint.js` (Quint), a JavaScript library that uses HTML5, SVG, and Web Audio and that provides a number of abstractions that support efficient creation of small interactive audio-visual programs ("applets") that can be embedded in other HTML content. Some implementation issues, namely cross-browser differences in handling SVG `foreignObject`, remain to be resolved.

We used Quint to make several applets that demonstrate various physical, acoustic, and psychoacoustic phenomena. The applets were used for teaching music technology within fine arts, and to support technical ear training for students of audio engineering.

Our current work aims mainly to improve and further develop the ear-training applets and extend their selection to other types of signal processing related to audio engineering. Students of audio engineering and other audio professionals would use the applets to practise and improve their listening skills in as many relevant areas as possible.

For future work we plan to use the applets in a larger context of integrated "non-linear" lecture notes related to audio arts. We hope that our pages may then be used by other educators and students in various learning environments.

5. REFERENCES

[1] Bachelor of Music – Digital Audio Arts. `http://digitalaudioarts.ca`.

[2] Hybrid pedagogy: a digital journal of learning, teaching, and technology. `http://www.hybridpedagogy.com`.

[3] A. Barker. *Scientific Method in Ptolemy's 'Harmonics'*. Cambridge University Press, 2001.

[4] M. Bostock, J. Heer, and V. Ogievetsky. *D3.js*: A JavaScript library for manipulating documents based on data. `http://d3js.org/`.

[5] I. G. Burleigh. *Quint.js*: A JavaScript library for building simple 2D machines in SVG. `http://quinta.audio/Quint/`.

[6] A. Case. *Mix Smart: Pro Audio Tips for Your Multitrack Mix*. Mastering music. Focal Press, 2011.

[7] J. Corey. *Audio Production and Critical Listening: Technical Ear Training*. Focal Press, 2010.

[8] H. Helmholtz. *On the sensations of tone as a physiological basis for the theory of music*. Longmans, Green, and Co., London, 1885.

[9] B. Katz. *Mastering Audio: The Art and the Science*. Focal Press, 2007.

[10] J. Meyer. *Acoustics and the Performance of Music*. Springer, 2009.

[11] D. Moulton. *Golden Ears*: An audio ear-training course for recording engineers, producers and musicians. `http://www.moultonlabs.com`.

[12] H. Partch. Barbs and broadsides. *Percussive Notes, Research Edition*, 18(3), 1979.

[13] M. S. Puckette. *Pure Data*: An open source visual programming language. `http://puredata.info/`.

[14] E. Sengpiel. Forum für Mikrofonaufnahmetechnik und Tonstudiotechnik. `http://sengpielaudio.com`.

[15] W. Sethares. *Tuning, Timbre, Spectrum, Scale*. Springer Verlag, London, 2005.

[16] M. Stavrou and W. Westbrook. *Mixing with Your Mind: Closely Guarded Secrets of Sound Balance Engineering*. Flux Research, 2003.

[17] E. Varèse and Alcopley. Edgard Varèse on music and art: A conversation between Varèse and Alcopley. *Leonardo*, 1(2):187–195, 4 1968.

[18] B. Vercoe, J. Fitch, V. Lazzarini, and I. Varga. *Csound*: A sound design, audio synthesis, and signal processing system. `http://www.csounds.com/`.