Termpot: criação e edição de funções no navegador em tempo de execução.

Guilherme Lunhani[1] **, Flávio Luiz Schiavoni**[2]

[1]Instituto de Artes e Design – Universidade Federal de Juiz de Fora
Juiz de Fora, MG

gcravista@gmail.com

[2]Departamento de Computação – Universidade Federal de São João Del Rei
São João Del Rei, MG

fls@ufsj.edu.br

Abstract. *This panel reports the development of a sound synthesis program, Termpot. The Web Audio API technology was used in a reinterpretation of a [Mathews and Moore 1970]'s work, GROOVE. The resulting work is still in its infancy, but allows create and edit sound functions at runtime, in a web browser.*
Keywords*: GROOVE; Web Audio API; DSP.*

Resumo. *Este painel reporta o desenvolvimento de um programa de síntese sonora, Termpot. A tecnologia Web Audio API foi usada em uma reinterpretação de um trabalho de [Mathews and Moore 1970], GROOVE. O trabalho resultante ainda é incipiente, mas possibilita criar e editar funções no tempo de execução, em um navegador web.*
Palavras-chave*: GROOVE; Web Audio API; DSP.*

1. Introdução

A síntese sonora em *web browsers* é sumarizada por [W3C 2012, Roberts et al. 2013, Wyse and Subramanian 2014].[Srikumar 2013] exemplifica uma concatenação de *nós de áudio* em um grafo de DSP. Três nós diferentes são representados na figura 1. Existe um outro nó (figura 2) que possibilita definir formas de ondas customizadas. O *Termpot* explora este nó, e delega, como o *wavepot* [1], as customizações para um(a) improvisador(a), que cria suas próprias *funções de transferência* e *interfaces de controle* [Mathews and Moore 1970].

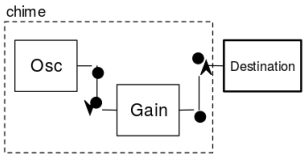

Figura 1: Estrutura básica de um sintetizador webaudio. *OscilatorNode*, *GainNode* e *DestinationNode* (alto-falante). **Fonte**: [Srikumar 2013]

2. Trabalho relacionado: GROOVE

Reinterpretamos um trabalho de [Mathews and Moore 1970], o sistema GROOVE, *Generated Real-time Operations On Voltage-controlled Equipment*. A compositora

[1]Disponível em https://www.github.com/wavepot/wavep0t

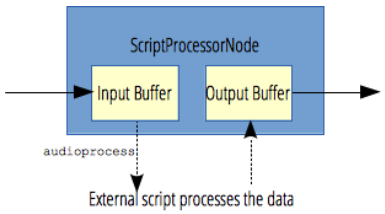

Figura 2: "A interface do *ScriptProcessorNode* permite a geração, processamento ou análise de áudio usando JavaScript". **Fonte**: [W3C 2012]

Laurie Spiegel (figura 3) sumariza suas as características, durante a produção de *The Expanding Universe* (1975)[2], entre as salas 2D-506 da Bell Labs (contendo o computador DDP-224) e a sala analógica 2D-562 (laboratório de Mathews):

> Todas as músicas no GROOVE eram representadas na memória digital como funções abstratas do tempo, séries paralelas de dois pontos, cada ponto sendo um instante no tempo e um valor instantâneo. A taxa de amostragem para essas funções, usada principalmente como controle de voltagem, era cronometrada por um grande e antiquado oscilador analógico que era normalmente fixado em 100 Hertz, cada ciclo do oscilador pulsando à frente do código, o computador lia, em cada uma das funções, naquele ponto do tempo, todos dispositivos de entrada e executava todas amostras. (...) Tínhamos uma pequena caixa com 4 potenciômetros e quatro chaves (alternadores fixados onde você os colocava) e dois botões de disparo. [3]

Figura 3: Laurie Spiegel configurando a saída analógica do GROOVE, durante a produção de *The Expanding Universe*. **Fonte**: [Spiegel 1975].

[2]Disponível em `https://www.youtube.com/watch?v=dYUZmsfm4Ww`.

[3]Tradução de *All music in GROOVE was represented in digital memory as abstract functions of time, parallel series of point pairs, each point being an instant in time and an instantaneous value. The sampling rate for these functions, which would be used mostly as control voltages, was clocked by a big old-fashioned analog oscillator that was usually set to 100 Hertz, each cycle of the oscillator pulsing one run through the code, the computer reading all of the real time input devices and playing of all of the samples at that time point in each of the time functions. (...) We had a small box with 4 knobs, 4 set switches (toggles that stay where you put them) and 2 momentary-contact push buttons on it.*

3. Metodologia de desenvolvimento

1) Customização um emulador terminal *Ptty.js* (`http://code.patxipierce.com/jquery-plugin/ptty/`). *2)* Definição de um ambiente interno, baseado no ambiente *Wavepot*, controlador do *ScriptProcessorNode*. *3)* Definição de comandos deste ambiente interno: inspeção de funções, definição de novas funções, tocar, parar, pausar, gravar e download, criação de controles gráficos (*jQueryUI*) e gravação (`https://github.com/mattdiamond/Recorderjs/blob/master/recorderWorker.js`).

4. Resultados

O *Termpot* (ver figura 4) [4] é uma customização do *Wavepot*. Funções sonoras podem ser definidas em linguagem *coffeescript* [Burnham 2011], executadas e gravadas (Código 1).

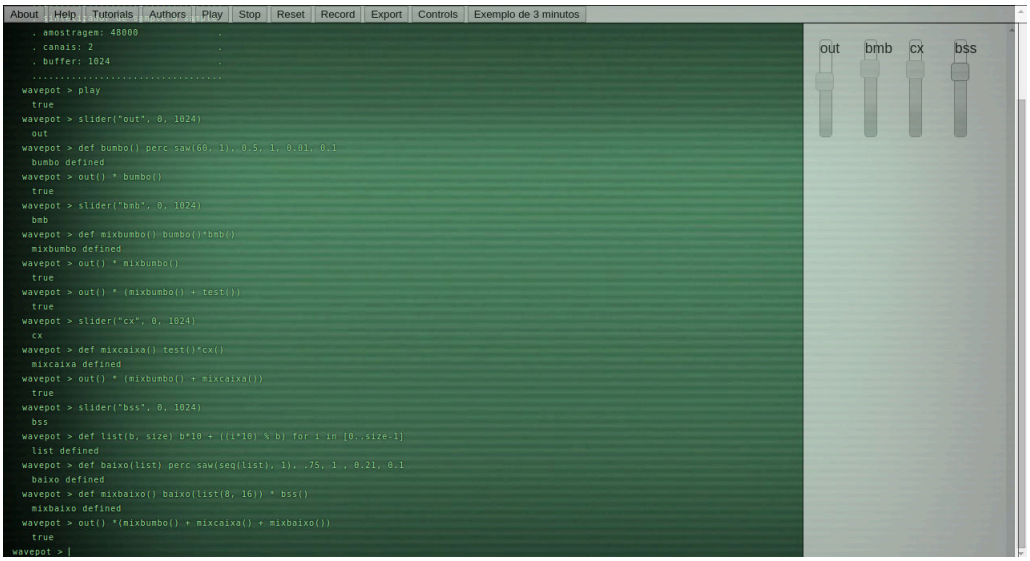

Figura 4: Aplicativo *Termpot*. **Fonte**: autores.

Problemas Técnicos

Existem *xruns*. Um *xrun* "(. . .) pode ser um estouro de buffer ou de uma saturação de um buffer. Um aplicativo de áudio ou não foi rápido o suficiente para transmitir dados (. . .) ou não rápido o suficiente para processar os dados"[User:Markc 2013][5]. Existe uma hipótese, não verificada, que a possível fonte dos *xruns* é a biblioteca *jQuery*.

5. Conclusão

O *Termpot* está em estágio incial de desenvolvimento. Com a reinterpretação de um caso histórico, desafios como síntese sonora, performance, e sintaxe simplificada em uma linguagem de programação, necessitaram de algum esforço. Por outro lado, existem problemas técnicos. Neste sentido, o desenvolvimento do *Termpot* aguarda contribuições da comunidade, com especial interesse em uma abordagem pedagógica para o ensino de processamento de sinais digitais de áudio na internet.

[4]Disponível em `https://jahpd.github.io/termpot`.

[5]Tradução nossa de *An "xrun"can be either a buffer underrun or a buffer overrun. In both cases an audio app was either not fast enough to deliver data (. . .) or not fast enough to process data.*

```
1  $ |                                                    (1)
2  $ wavepot 1024                                         (2)
3  ..................................
4  . sintetizador de sample a sample.                     (3)
5  . amostragem: 44100              .
6  . canais: 2                      .
7  . buffer: 1024                   .
8  ..................................
9  wavepot > play                                         (4)
10 true
11 wavepot > def AM440(f, a) sin f, sin(440, a)           (5)
12 AM440 defined
13 wavepot > inspect                                      (6)
14 funcoes definidas
15 tau  tmod  mute  stereo  sin  sin2  saw  ramp  ttri
16 tri  sqr   pulse noise         perc test  seq  bpm   nextevent
17 AM440
18 wavepot > slider "f", 1, 1025                          (7)
19 true
20 wavepot > slider "a", 0, 1024
21 true
22 wavepot > record                                       (8)
23 true
24 wavepot > AM440 f()*1000, a()                          (9)
25 true
26 wavepot > export                                       (10)
```

Código 1: Ptty aguardando dados de entrada do improvisador (1). *Boot* do ambiente *wavepot* com um buffer de 1024 amostras por ciclo de DSP (2). Informações diversas do sistema (3). Execução do DSP (4). Definição de uma função *AM440* (5). Informações sobre as funções diponíveis (6). Definição de GUIs (7). Gravação em um arquivo de áudio (8). Execução da função *AM440* com controles (9). Download da gravação (10)

5.1. Trabalhos Futuros

i) Criar um servidor; *ii)* otimizar o emulador, talvez substituindo o Ptty ou propondo melhorias; *iii)* suporte para amostras pré-gravadas.

5.2. Agradecimentos

Guilherme Rafael Soares pelas sugestões, aos desenvolvedores do *Wavepot*, e a FAPEMIG por subsidiar a pesquisa.

Referências

Burnham, T. (2011). Coffeescript: Accelerated javascript development. *Pragmatic Bookshelf*.

Mathews, M. V. and Moore, F. (1970). GROOVE a program to compose, store, and edit functions of time. *Bell Telephones Laboratories*, page 7.

Roberts, C., Wakefield, G., and Wright, M. (2013). The web browser as synthesizer and interface. page 6.

Spiegel, L. (1975). The expanding universe: 1970s computer music from bell labs by laurie spiegel. disponível em `http://www.retiary.org/ls/expanding_universe/`. *Retiary*.

Srikumar, S. (2013). Tamming the script processor node. disponível em `http://sriku.org/blog/2013/01/30/taming-the-scriptprocessornode/`.

User:Markc (2013). Xruns: From the alsa wiki. disponível em `http://alsa.opensrc.org/Xruns`.

W3C (2012). Web audio API.

Wyse, L. and Subramanian, S. (2014). The viability of the web browser as a computer music platform. *Computer Music Journal*, 37(4):10–23.