

Web Audio Modules

Jari Kleimola

Dept. of Computer Science
Aalto University
Finland

Oliver Larkin

Music Department
University of York
UK

outline

background

WAM API

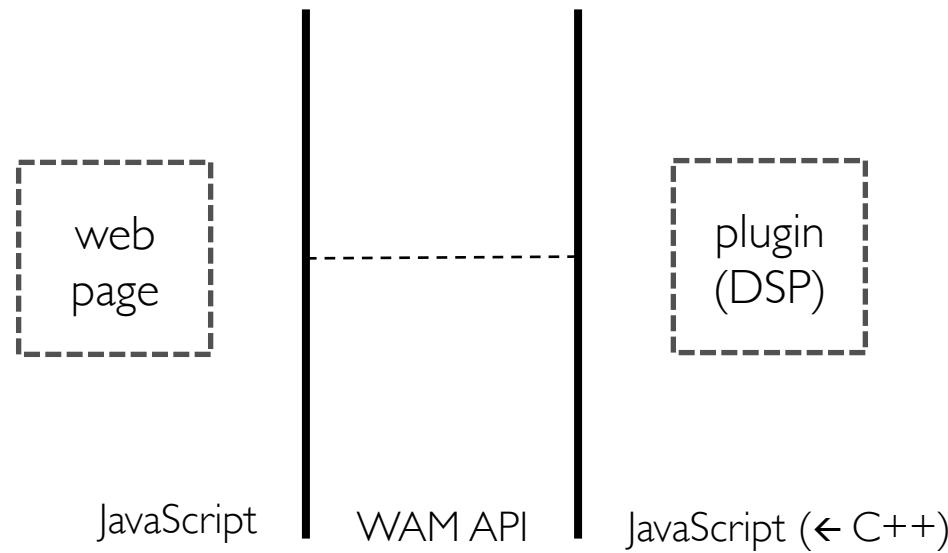
demo

implementations

conclusion

intro

browsers | DAWs
web audio modules | plugins
web browser affordances + music making, performing



native plugin apis



Audio Units



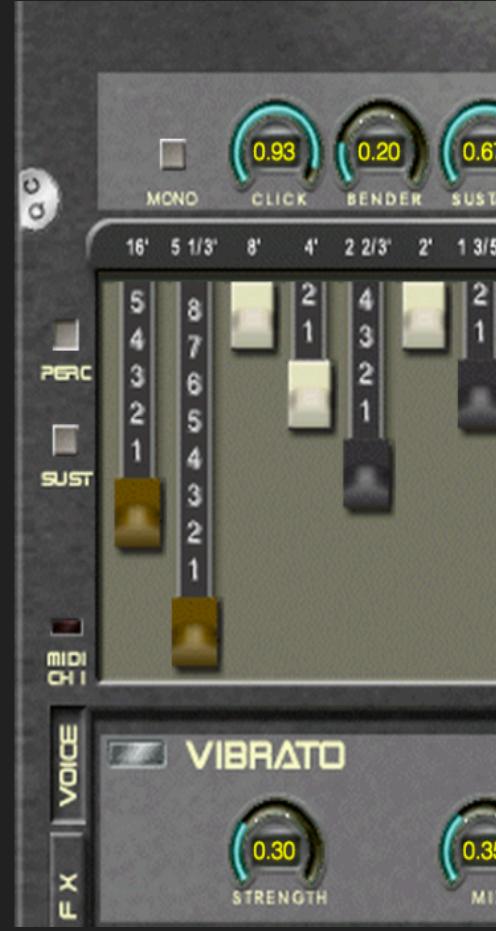
LADSPA

DSSI



DAW Plugins for Web Browsers

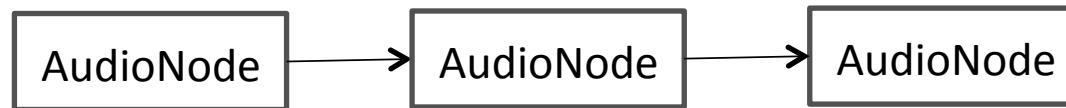
Jari Kleimola / WAC 2015



<https://mediatech.aalto.fi/publications/webservices/dawplugins>

web audio

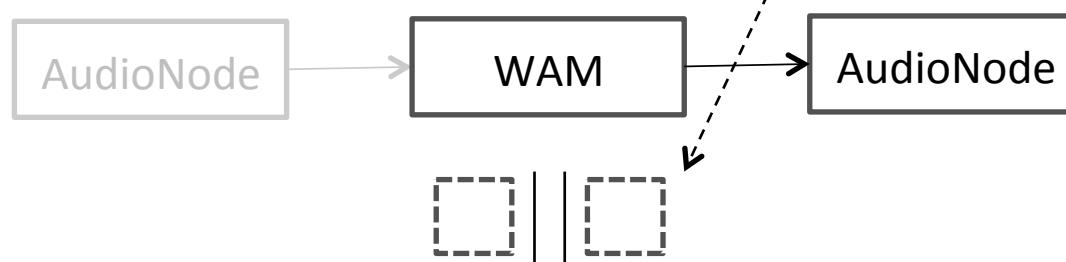
web audio api



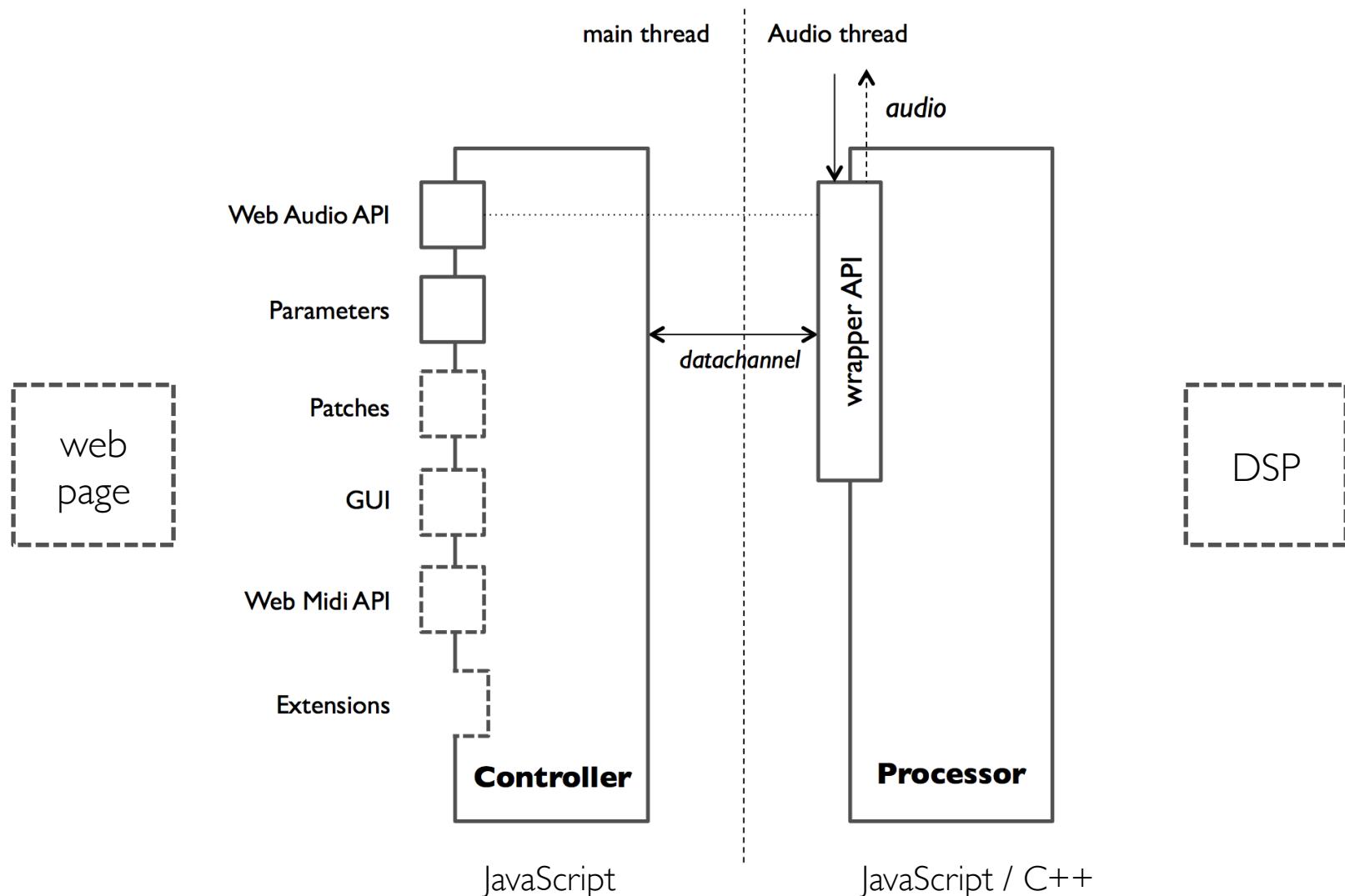
cross-compilation



web audio modules



wam api



demo

WAMs SYNTHS EFFECTS DEVELOPERS BLOG

Web Audio Modules

COMMUNITY SITE

synthesizers and audio effects processors for web browsers

BROWSE SYNTHS

or scroll down to learn more

wam usage

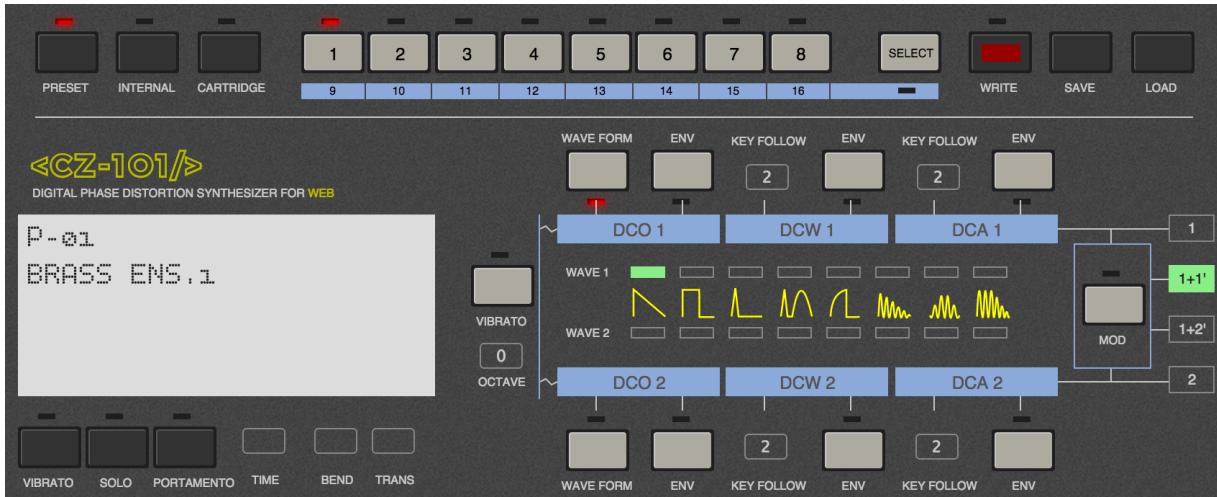
include into a webpage

```
<link rel="import" href="http://webaudiomodules.org/synths/wam-webdx7.html"/>
<wam-webdx7 autoconnect></wam-webdx7>
...
var dx7 = document.querySelector("wam-webdx7").controller;
dx7.postMessage([...]) etc.
```

create one yourself

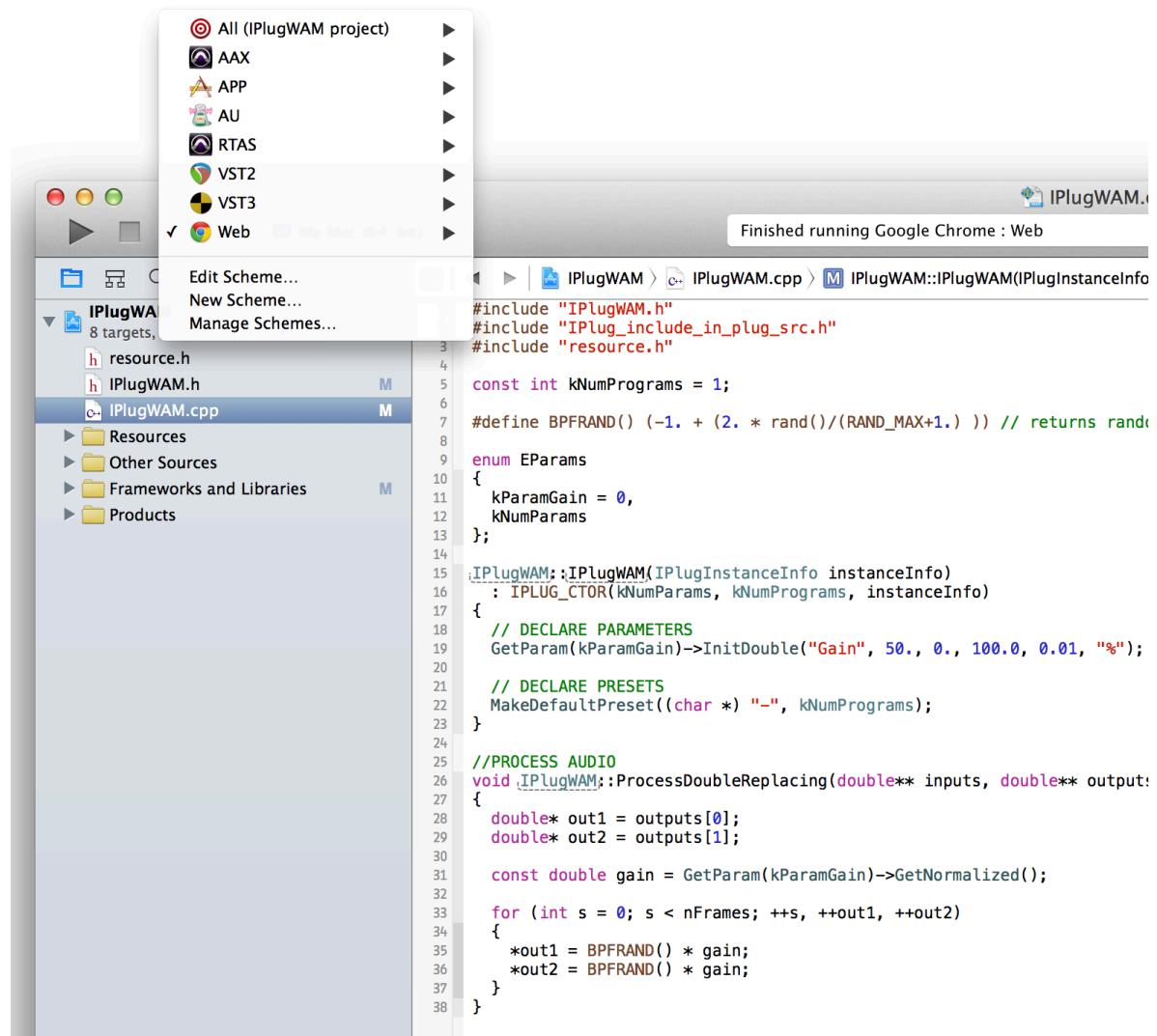
- inherit WAM.Controller
- inherit WAM.Processor / WAM.Synth
- or (re-)use C++ code

webCZ-101/VirtualCZ



IPlug Web Target

- Export IPlug C++ code to ASM.js with template controller JS + HTML glue
- Currently plug-in GUI needs to be re-written



The screenshot shows the Xcode interface with the IPlugWAM project open. The left sidebar displays the project structure with targets: All (IPlugWAM project), AAX, APP, AU, RTAS, VST2, VST3, and Web (which is selected). Below the targets, files like resource.h, IPlugWAM.h, and IPlugWAM.cpp are listed. The main editor window shows the IPlugWAM.cpp file with the following code:

```
#include "IPlugWAM.h"
#include "IPlug_include_in_plug_src.h"
#include "resource.h"

const int kNumPrograms = 1;

#define BPFRAND() (-1. + (2. * rand()/(RAND_MAX+1.)) ) // returns random float between -1 and 1

enum EParams
{
    kParamGain = 0,
    kNumParams
};

IPlugWAM::IPlugWAM(IPlugInstanceInfo instanceInfo)
    : IPLUGCTOR(kNumParams, kNumPrograms, instanceInfo)
{
    // DECLARE PARAMETERS
    GetParam(kParamGain)->InitDouble("Gain", 50., 0., 100.0, 0.01, "%");

    // DECLARE PRESETS
    MakeDefaultPreset((char *) "-", kNumPrograms);
}

//PROCESS AUDIO
void IPlugWAM::ProcessDoubleReplacing(double** inputs, double** outputs)
{
    double* out1 = outputs[0];
    double* out2 = outputs[1];

    const double gain = GetParam(kParamGain)->GetNormalized();

    for (int s = 0; s < nFrames; ++s, ++out1, ++out2)
    {
        *out1 = BPFRAND() * gain;
        *out2 = BPFRAND() * gain;
    }
}
```

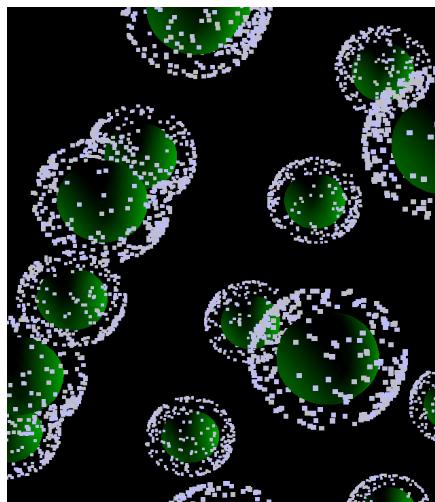
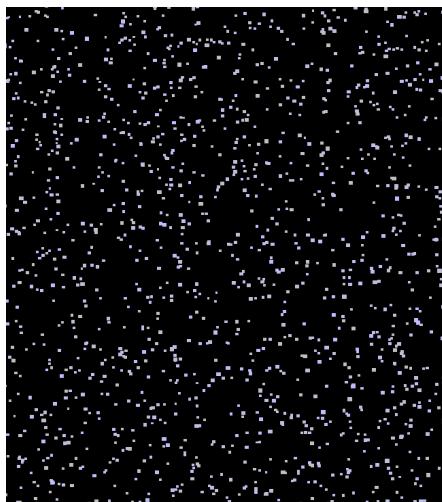
webDX7

virtual Yamaha DX7 synthesizer

- sound engine by Raph Levien
- ported from vanilla C++, GUI took more time than DSP

patchcloud

- lots of patches in the internet
- interactive webGL visualization



evaluation

Latency

WAM	default	buffer = 32
webCZ-101	48 ms	33 ms
webDX7	45 ms	31 ms

Polyphony

WAM	asm.js	native	
webCZ-101	60	200	30%
webDX7	17	128	13%

commercial issues

- Relatively easy to add support for WAM API to an existing C++ codebase
 - Multithreading, explicit SIMD and platform specific ASM might cause problems
 - GUI needs reworking (but Web UI is the way interfaces are going anyway)
 - Performance slower than native
- emscripten's ASM.js code browseable, and algorithms viewable
- Wams are easily shareable

future work

hosting in soundation

PNaCl + Web Assembly

AudioWorkerNode

GUI codebase reuse

<http://chrome.soundation.com>



conclusion

proposed streamlined Web Audio Modules API

- virtual instruments and audio effects processors for web browsers
- load with the rest of the page (no manual installation)

two proof of concept implementations

- latencies higher than in native, but gap is narrowing

community website

- hosts WAMs
- documentation
- feedback

THANKS FOR LISTENING !

webaudiomodules.org

jari.kleimola@alumni.aalto.fi

oliver.larkin@york.ac.uk