Common voice - WebAssembly MP3 encoding

moz://a

Mentor - Gregor Weber

Personal Details

- Name: Mritunjay Goutam
- Email: mritunjaygoutam2204@gmail.com
- IRC nick: mritunjay
- Telephone: +91-7889387127
- Other contact methods: Twitter -> mritunjay_mohit, Slack-common-voice-> mritunjay
- Country of residence: India
- Timezone: GMT +5:30
- Primary language: English, Hindi

Project Proposal

Common voice - WebAssembly MP3 encoding

Current state and problem

```
var sourceNode = audioContext.createMediaStreamSource(microphone);
var outputNode = audioContext.createMediaStreamDestination();
this.recorder = new MediaRecorder(outputNode.stream);
```

This is how right now our project is using MediaRecorder to manipulate and record output stream of PCM data from audioContext source to play them instantly in web. Due to this missing MediaRecorder API in several browser platforms like Safari and Edge voice-common is not supported for Safari and Edge.

Missing MediaRecorder API leads to a problem in:

- capturing the data generated by a MediaStream object for analysis, processing or saving to disk.
- Examining and controlling the recorder status.
- We have to perform manual encoding operations on data according to browser format support.

- ## Research, work, and discussion

For overcoming this missing API we have to manually store the media source's stream as a chunk and then convert them in blob URL (file-like object of immutable, raw data) of type audio/wav. After this, we will design a WASM compiled function from any native mp3 encoder which takes blob URL as input (no need to pass bitRate, and output pointer we can hardcode that in WASM function) and it returns a file or buffer stream (as data URI) as output.

Some native mp3 encoders -

- fdk-aac (advance audio coding) : As of now, this is the lightest and fastest mp3 encoder which is very well defined and easily written.

  Tested - It takes 3.1s to encode 9.5 mb WAV file to mp3

- lame : LAME is considered the best MP3 encoder at mid-high bitrates. Quality and speed improvements are still happening, probably making LAME the only MP3 encoder still being actively developed.

  Tested - It takes 5.6s to encode 9.5 mb WAV file to mp3

- ffmpeg : ffmpeg can be the first choice if we can make its libfdk-aac encoder compile separately. Otherwise, it is a whole bunch of extra feature that we do not need.

Technical Milestone

- C to WASM
- Integration to javascript browser audio (or worker)

C to WASM

Unlike compiling a single c file, building project with Emscripten require some extra effort like writing makefile or we can manually build it by passing argument `./emconfigure ./configure` and `./emmake make` while building project for linking C program header file's bitcode to project's source folder for including the header files.

(***Note** - I have compiled the one of three proposed encoder fdk-aac for an example here is the Repository)

In the case of fdk-acc and ffmpeg we have to install libfkk-acc because encoder uses them as system included library `#include <fdk-aac/aacenc_lib.h> or /path-to-des.`

1. After selecting for perfect native encoder first we have to build the projects with Emscripten by configuring makefile. Otherwise, we have to manual pass all the multiple file to **emcc** compiler to Compile the **linked** header file's bitcode generated

by make. Right now our linkage is like `libfdk-aac --> fdk-aac --> ourCustomCode.c` . They all are connected with `#include` in their header. Example emcc command is `emcc main.c -libfdk-acc <path to libfdk-aac> /src -o lib.out`

2. Once the project builds, we have to start writing our C programme that should expose our custom function that we have to use in our javascript and remove main default function. Send raw byte to encoder or we can try virtual file system for WASM (for blob URL as input and treat it as file object) by passing `-s FORCE_FILESYSTEM=1` but here again we loose one opera 12.16 which has limited support for the W3C File API. Once we pass raw byte as input instead of file pointer(blob), we can delete `fscanf` function and set `input pointer` to buffer. Our pseudo C program should be like

```
#include <emscripten.h>

#include <fdkaac/src/aacenc.h>

EMSCRIPTEN_KEEPALIVE

string customEncoder(uint_8_t *buf, size_t len) {

//no main

//no fscanf for file

//this is default encode function

encode(aacenc_param_ex_t *params,pcm_reader_t *reader,HANDLE_AACE
NCODER encoder, uint32_t frame_length,m4af_ctx_t *m4af);

//IMPORTANT - this is our modified costum encode function which
we need to call inside our defined function

encode(pcm_reader_t *reader); //reader is pointer to our raw data

                              // and other arguments can be hardcoded

}
```

Inside encode function we also do not need `fwrite(frame->data, 1, frame->size, ofp);` We only return `frame->data` which our function coustomEncoder return to javascript calling instance.

3. Recompile with command `emcc -s WASM= 1 -s \ -s EXPORTED_FUNCTIONS= "['coustom Encoder']" \ -s EXTRA_EXPORTED_RUNTIME_METHODS='["ccall"]' \-o`

`audioplayer.html \ fdk-aac/src/main.c` . This cmd will expose our customEncoder function to be used in javascript with `Module.ccall` (we can also use cwrap). Above command will also create audioPlayer.html which include our WASM js output file through script tag. Now testing of compiled function with audioPlayer.html :

- Input-based: we have check whether giving raw data as input instead of blob URL is working or not. And in a case, we should also try the blob URL option as input file with enabling virtual file system for WASM.
- Output-based: If we succeed with output as raw data buffer then great else we can pass the output file path of our local project directory with clip number as a name. So here one more input we have to send to our function that is clip number and has to generate output filename with clip number.

4. **Make NPM module for WASM mp3 encoder :** we can bind WASM exposed function for js by including a.out.js `Module.onRuntimeInitialized = _ => { const encode = Module.cwrap('encode','string', ['string']);`

   // pass the returned data to worker thread and postmessage(blobURL) of {type: "audio/mpeg"}};

5. **Integration of encoder to javascript browser audio :** At this point we can get audio buffer containing PCM data and send it to above describe NPM module for WASM encode function and in return we get blob url. It would be like:

```
var audioCtx = new AudioContext( );
var scriptNode = audioCtx.createScriptProcessor(4096, 1, 1);
scriptNode.onaudioprocess = function(e){
//data = e.inputBuffer.getChannelData(0)
//pass the data to WASM encode function
// blobUrl = encode(data, clipnumber);
}
```

Post GSOC :

I will work on setting Voice Wave Avatar based on the user's unique voice (1591). Try to extract and set points for all phonemes with the help of PocketSphinx.js or Google cloud speech interface.

Benefits of the project

- Support for recording and playback it instantly features for all browser with missing media Recorder API and fixes #469 a very waited and important issue. We can do away native app.
- We can have our NPM module as a library for WASM based mp3 encoder. We can set an example for a solution to how missing APIs can be filled with native software.

Schedule of Deliverables

21 April - 10 May	Busy in semester exams. Arranging my stuff ready for summer vacation.
10 May - 27 May (Community bonding)	Discuss with mentor to choose exactly one native encoder to be used for webAssembly. I have mentioned path and way for the solutions in the proposal but try to modify it as per the common view with the mentor. Along with this, I will learn the necessary thing which can help me to make the solution more easier and accurate.
27 May - 14 June	After encoder choice complete. We start writing a makefile for WASM build. And complete milestone 1.
15 June - 26 June (Evaluation 1)	This is time is to write our C programme for our encode function. Complete Milestone 2.
28 June - 12 July	Here the testing part is to be done after WASM compiled js and HTML output. Complete Milestone 3.
13 July - 22 July (Evaluation 2)	Designing it and publishing the encoder on NPM. complete Milestone 4.
23 July - 8 August	Integration of above encoder module with the voice-web project. Complete Milestone 5.

8 August - 18 August	Ask mentor something else should be done along with these. And write a good blog post with summarizing all my work.
19 August	Final Evaluation Done.

Work/Internship Experience

Google Summer of Code Student 2017
Organization - CMUSphinx
I work on speech recognition and mobile browser-based 3D animation for lip syncing for phonemes.

At Viithiisys Technology (May - Jul) 2018
Project named Refrase which is AI-based job searching portal I worked in implementing Elastic search API and frontend of Refrase. Appraisal letter - Appraisal

Website for University tech fest
http://titiksha.smvdu.ac.in/#

Open Source Development Experience

Mozilla - Voice-web - From last couple of months I have contributed to the voice-web project. Now I have a good grip of voice-web codebase from the server to the client. You can get my open and closed PR Here

Nodejs - I have contributed some patches in nodejs core as a test, build script and docs related.

Vue burger menu - I am a contributor to this burger menu theme based on Vuejs. I worked on adding some menu opening effects like Bubble and Falldown. Also some small fixes.
Remote-host - A web-based tool for connecting to a remote system using SSH. Due to its web-based, we can use this client SSH logging application on any platform. Just install it via `npm install -g remotehost`

Owncloud - Client dashboard for owncloud new js based API like cern-box.

My resume can be referred for more details about my contributions and work experience.

Academic Experience

I am currently in the pre-final year of my Bachelor in computer science degree. I learned C as academic syllabus and JS is a self-taught skill. I have been programming from last 4 years. Since from my first year, I started contributing to open source and till date, I learned a lot of programming and open source.

Will I submit a proposal to other organization?

No, I will only submit proposal for voice-web Mozilla

Why Me

I have a decent knowledge of the workflow of web audio API due to my interest in speech recognition. I want to be a long term maintainer of this project. Now I have well sync with my mentor working time. I am familiar with the codebase. I can complete the project in given time.

Why Mozilla

Mozilla is one of the best open source organization for you to be a part of as a contributor. And it is more reasonable when the project idea is very exciting, of your interest and related to modern web technology like WebAssembly. Mentors in Mozilla are very welcoming and responsive so, it helps you in your learning. Through Mozilla, I can show my project solution of WebAssembly to a large audience. And adding Mozilla in my work portfolio itself a great achievement.