Online Audio Editor in Freesound

Roberto Pérez Sánchez
Universidad Pompeu Fabra
robertops1818@gmail.com

Frederic Font Corbera
Universidad Pompeu Fabra
frederic.font@upf.edu

ABSTRACT

Freesound has a huge community dedicated to upload and share sounds online for other users to use those sounds in their productions. In some cases, the sounds will need to be transformed and users will have to go in their devices and edit that sound in order to get the exact version that fits their production the most. In this paper, an audio editor is developed and added into the Freesoud environment in order to have those two steps together in the same workflow. This approach will add a huge value to the Freesound experience and will differentiate it from its competitors. For the development of this editor, different Javascipt and HTML frameworks have been used, like Wavesurfer (based on Web Audio API) or Bootstrap, always trying to keep the interface and functionalities accessible and easy-to-use to a general audience. Finally, two evaluation methods have been developed, the first in order to get some general feedback about the use of the editor and the second one to be sure that all its different functionalities behave as expected. The first evaluation showed the need of a simpler interface and functionalities, which was developed and then tested using the second evaluation approach, showing that all different actions, filters and effects worked as expected.

1. INTRODUCTION

Looking at the numbers presented by Font in [1], the number of sound downloads in Freesound during 2019 was around 20.6 million. That is 1.6M more than the previous year. Having this huge number of downloads, which is continuously increasing every year, the website would benefit from a tool to let users take the most advantage of the sound they want to download by selecting only one of its parts or applying some effects and filters to them, simplifying their workflow by not having to use another platform like Audacity to edit the sample. In this paper, an online embedded editor in the Freesound web will be developed and tested in order to see if this tool will add some extra value to the community experience while using the system.

First of all, we will present an overview of the state of the art regarding Freesound, the existing stand-alone editors in the market, different usages and frameworks of the Web Audio API and some evaluation techniques commonly used for testing these types of projects. Then, the methodology followed for the development of the editor will be explained, taking into account the implementation of the code and the evaluation methods used (usability questionnaire and user cases). Finally, we will present the different versions of the editor during its development, outlining the different changes that occurred in the interface when performing the evaluations, as well as some conclusions to sum up how Freesound will benefit from the addition of this editor, and possible lines of future work regarding its development.

2. STATE OF THE ART

2.1 Freesound

Since its creation in 2005, Freesound[1] has developed a huge user community and sound database. In 2012, as stated in [2], there was more than 2 million registered users in the platform and more than 140,000 uploaded sounds. Nevertheless, Font et al. also noticed that 89% of these users are considered as 'regular' users, which are the ones that have registered to the site but never contributed to it, maybe just registering in order to download some samples, and less of 1% of the total users are the ones that upload sounds. Although this gives the impression that there are not many features that engage the user to participate actively in the system, either uploading sound or participating in forums, the reason for that low percentage of contributors could be explained because it requires the user to have some related skills and (possibly) gear. Adding these new features, for example an embedded editor in the web, might make the platform more attractive both for contributors and for consumers.

Nowadays, Freesound counts with more than 8 million registered users and more than 400,000 sounds and effects available to be downloaded. This growth of users and sounds have led to the development of new features in the main website and subprojects, for example, Freesound Datasets (later renamed as Freesound Annotator). Fonseca and his colleagues explain in [3] the use of Freesound samples in order to create open audio datasets based on four principles: transparency, openness, dynamism and sustainability. Regarding these new features or subprojects related to the platform, a new "idea" appeared: an audio editor embedded on the website in order to edit the sample prior to its download. This new feature will give the user the opportunity to perform quick fixes of the sample, as well as applying different effects and filters to it without having to execute a different program just for editing the sound. From here, it derives the research question of this paper: **How an easy-to-use, maintainable, and efficient audio editor embedded in the Freesound website could be developed?**

[1] https://freesound.org

Web Audio Conference WAC-2021, July 5–7, 2021, Barcelona, Spain.

2.2 Web Audio API

The goal of this project is to implement such audio editor. In order to develop and insert the editor into the Freesound website, the Web Audio API[2], a modern and powerful audio framework for the browser, will be used. Historically, this API was not the first approach in order to work with audio in the web. As Boris Smus explains in his book 'Web Audio API' [4], the first way of playing sounds on the web was using the <bgsound> tag in HTML, but it was very limited as it only allowed to play background audio in the browser. More recently, the <audio> tag was the one used in order to perform these tasks until the creation of the Web Audio API in late 2011, which was created in order to give a better performance of those tasks, as well as giving them a bit more flexibility, bringing a fully featured audio framework to the browser.

Frequently, this API is wrapped into other frameworks to make it easier and intuitive to work with it. Some frameworks studied for the development of this project are the following:

- Wavesurfer.js:[3] Customizable audio waveform visualization JavaScript framework.
- Pizzicato.js[4]: Web audio JavaScript library that aims to simplify the way you create and manipulate sounds via the Web Audio API.
- Tone.js[5]: Framework for creating interactive music in the browser. It provides advanced scheduling capabilities, synths and effects, and intuitive music abstractions built on top of the Web Audio API.

Taking this into account, the development of an easy-to-use, maintainable and efficient web audio editor should be very feasible. On top of that, it would add a huge amount of value to the Freesound webpage, because, to the best of my knowledge, similar websites like FindSounds[6] or FreeMusicArchive[7] do not feature an audio editor like the one being envisioned in this project.

2.3 Existing editors

Some of the most relevant features that an easy-to-use audio editor should give to the user are, according to [5], (1) simple and uncluttered workflow and interface, (2) use of effects in order to alter the sample and (3) support for a wide array of audio formats (WAV, MP3, etc.). Taking these requirements into account, some of the most relevant ones which can resemble more the idea of the web audio editor for Freesound are the following:

- Audacity. It is the most capable free audio editor. It provides the user with a full set of editing and mastering tools, as well as a pretty complete list of audio effects, sound generator plugins and export formats.
- Ocenaudio. It is a free single-track editor for making destructive edits to audio files. Destructive edit means that the changes are made in the same sample file whenever the user hits the 'Save' button. This destructive approach is the one also used in Audacity, but Ocenaudio gives a clearer interface and includes the basic operations that Audacity is capable of performing.
- TwistedWave[8]. This editor offers two options: a stand-alone desktop application or an online web-based audio editor with the same functionalities. It can be used to create recordings directly on the web or edit files you already own by just uploading them to the web. It also includes some effects on top of the usual editing features, like amplify audio, create fades and change pitch or speed, as well as applying different external VST effects. Within the free tier of TwistedWave, you can edit mono files up to 5 minutes long and create an account in order to keep your files there.

Apart from TwistedWave, there exist more simple online audio editors with very similar features between them. According to [6], TwistedWave is one of the top 5 online edition tools available on the Internet, but there are more, for example, Beautiful Audio Editor[9], Sodaphonic[10], Bear Audio Tool[11] or Hya-Wave[12]. Out of these five, TwistedWave is the only one supporting all available web browsers, since the others only work on Chrome and/or Mozilla. Despite this fact, the one that appeals to us the most due to its simplicity and performance is Hya-Wave. It is important to be aware that, to the best of our knowledge, neither TwsitedWave nor Hya-Wave are open-source projects, that means, their code and functionalities are not publicly available.

However, there exists another very capable online editor called AudioMass[13]. In this case, the whole project is open source, with its code available in Github[14]. This editor uses the previously mentioned Web Audio API framework Wavesurfer.js, which gives a lot of facilities in order to draw the sample waveform as well as methods which encapsulate the basic Web Audio API operations. AudioMass offers the basic playback controls for the sample and a catalogue of effects like fades, compressor, delay, distorsion or EQ among others. With some personalization and because it is open source, this editor can be embedded into other websites.

Hya-Wave provides a really easy-to-use and clear interface where the user can see all the features available in the editor. It provides the basic operations to cut, paste, crop or export the audio, but it also includes the possibility to record audio directly from the browser and to add up to 18 different effects to the sample, like amplification, fade in and out, low and high-pass filters, etc.

Out of the online audio editors presented previously, Hya-Wave is the one that has the most features similar to the ones that we want to develop in the web editor embedded in the Freesound website. Our idea is to handle the user the basic operations for formatting the audio and also offer the possibility to add different effects to the sample. We personally prefer Hya-Wave over AudioMass due

[2] https://developer.mozilla.org/es/docs/Web_Audio_API

[3] https://wavesurfer-js.org

[4] https://alemangui.github.io/pizzicato/

[5] http://tonejs.github.io/

[6] https://www.findsounds.com

[7] https://freemusicarchive.org/search

[8] https://twistedwave.com/online

[9] https://beautifulaudioeditor.appspot.com

[10] https://sodaphonic.com

[11] https://www.beraudiotool.com/sp/

[12] https://wav.hya.io/#/fx

[13] https://audiomass.co

[14] https://github.com/pkalogiros/audiomass

to its simplicity and interface, but being open source, the code of the AudioMass editor will be taken into account in the development of the Freesound one in order to get some inspiration and solve any problem that we could encountered.

2.4 Evaluation

Finally, one last thing that we have to have in mind when developing this system is the way that the editor and its functionalities are going to be tested and evaluated by final users in order to get some impression and thoughts. Taking a look at the Audio Commons[15] deliverable reports [7, 8], it seems that the evaluation methods for technological projects consist of different steps:

- Participatory methods. Prior to the development, it is a good practice to gather data from potential users regarding the features or functionalities that those users expect the system to have. These opinions should give the developer a good overview of which features to add to the system, as well as maybe some approach that he or she was not able to come across.
- Hands-on methods. After the development of the system, another set of users are asked to use the system and write down their opinions on the functionalities of it: is it easy to use, is it useful, does it have the functionalities that this type of system needs to have, etc.
- Informal methods. The developers can also retrieve useful information about the system when talking to potential stakeholders during demonstrations or performances

In conclusion and taking all of this information about the Freesound environment and the technologies involving the Web Audio API that are going to be used, as well as the common features and operations that other editors include and the evaluation methods that are going to be developed for the testing of the editor, all the necessary information in order to start the development of the editor has been retrieved and studied.

3. METHODOLOGY

3.1 Implementation

From the beginning, the structure of the project has been the following: HTML file containing the visual elements of the interface, a CSS file for customization of the elements and several Javascript files containing the logic of the editor. Both the code of the stand-alone editor and the Freesound integrated editor are available in my Github account as public repositories[16] [17].

3.1.1 Wavesurfer

In the State-of-the-Art chapter, different Javascript frameworks oriented to audio treatment were studied, but the one which is finally used is Wavesurfer. Mostly all the functionalities of the editor are based on the different methods available in its API, apart from some of them where other plugins (PitchShifter, SmoothFade or Bitcrush) or just plain Web Audio API code are used

The first thing that was created was the waveform visualization. For this, a tag is created and named after a unique ID. Then, in the main JS file, a Wavesurfer instance is created referencing the ID of the HTML element. When this is done, the waveform is shown on screen, and the Wavesurfer object is ready to be used in order to execute the different methods available in its API.

At the time of creating the Wavesurfer object, the different plugins that are going to be used in the editor are also added to it. In this case, the three plugins used are: cursor (to move it throughout the waveform), regions (to select region in waveform for deleting or clearing) and timeline (to see the duration and actual time of the sample). Moreover, all the configuration parameters of the object are also configured, and the sample loaded into the object using the 'load' method.

Once the Wavesurfer object is created, it can be used in order to implement the different actions, filters and effects of the editor.

3.1.2 Functionalities

All of the functionalities of the editor are stored in a JS file (app.js). This file includes all necessary object initializations and methods declarations for the editor to work as expected. It also links the actions done in the UI (like clicking a button) with the method that has to be triggered when doing that action, via HTML query selectors and listeners.

3.1.2.1 Actions

The different available actions are separated in four panels:

- Playback controls. This includes play/pause, stop (returns the cursor to the start of the sample), go backwards and advance forward (both set to 2 seconds)
- Undo and redo workflow, as well as clearing and deleting regions of the sample
- Zoom panel: zoom in, zoom out and zoom to selected region
- Crop and get the selected region of the sample, get the original one, reverse the sample, download it and the help button, which opens a modal with details and shortcuts about the editor.

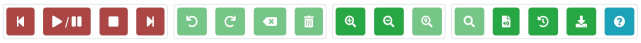

Figure 1. Editor actions.

3.1.2.2 Waveform visualization and timeline

The user can select regions on the waveform in order to perform some of the actions. Also, the cursor moves along with the sound and a visual timeline is displayed at the bottom.

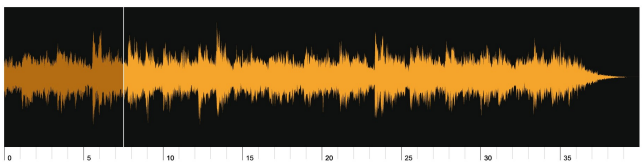

Figure 2. Waveform and timeline.

3.1.2.3 EQ panel

It contains the knobs for applying the three filters available: lowpass, bandpass and highpass. For the lowpass and highpass

[15] https://www.audiocommons.org/materials/

[16] https://github.com/robertops18/audio-editor-js

[17] https://github.com/robertops18/freesound

filter, only the frequency cutoff can be set, but for the bandpass filter, the user can also change the resonance factor Q.

3.1.2.4 Effects

The effects for this editor are all set using knobs. The available effects are:

- Gain control. The user can change the gain value of the sample from -20 to 20 dB.
- Fade in and out. When using these effects, the gain or the sample will increase from 0 to 1 at the start of the sample for the fade in and decrease from 1 to 0 in the case of the fade out. The time for this change is set using the knobs.
- Playback rate. The user can change the playback rate of the sample, also changing its pitch.

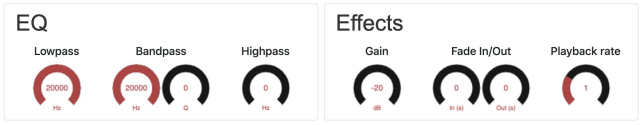

Figure 3. EQ and effects panel.

3.2 Evaluation

3.2.1 Usability questionnaire

After a first stable version of the editor was available, we created a simple questionnaire[18] in order to store some feedback and opinions about the style, performance and usability of this first version. With that information, we will be able to fix the different bugs that may appear in the functionalities of the editor, as well as taking into account the usability and UI suggestions made by potential users of the editor.

3.2.2 User cases evaluation

For a second evaluation, the performance of several user cases will be needed by some volunteer potential users of the application. These user cases will include different tasks to be done using the editor, as well as its expected behavior. Taking into account this expected behavior, users will need to take notes about the workflow they are coming across with and compare it with the expected one. This type of evaluation will give me deeper knowledge about the right or wrong behavior of all the functionalities of the editor. All of them follow the same structure. The complete user cases spreadsheet is available in Google Drive[19]

4. RESULTS

In the case of the general usability of the editor, users agree that the behavior of the knobs was not the appropriate one when performing the first evaluation. In the first version of the editor, the user would have to select the value in the knob and then apply the filter or effect pressing a button. The preferred workflow was to apply the desired filter changing the value of the knob, in order to experiment easily with them. This approach was developed in the second version of the editor, as well as the addition of tooltips for the actions, which was another feature requested by a user.

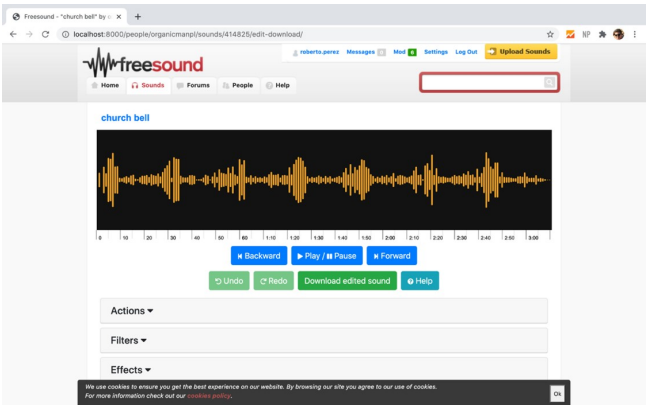

Figure 4. First version of the editor.

Regarding the interface, the general opinion was that it took a lot of space in the web. The knobs for filters and effects, and the action button were too big, and the user had to scroll through in order to apply them (Figure 5). In the second version, the user interface was completely changed in order to get all the editor functionalities just in one screen: the actions buttons were replaced by icons at the top of the waveform, some filters were deleted, and all knob's size decreased (Figure 6).

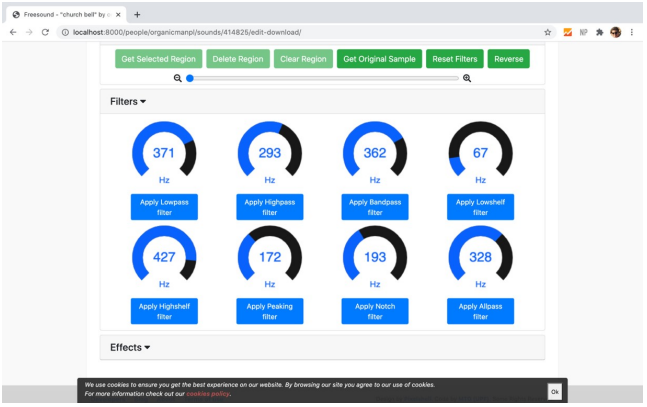

Figure 5. First version: filters dropdown.

Finally, the general impression about the editor was that it would definitely be useful in the context of Freesound, adding huge value to the user experience in the web. However, the editor needed some changes like the mentioned above, and some other functionalities like being able to change the playback rate (developed in second version) in order to be really useful.

---

18 https://docs.google.com/forms/d/e/1FAIpQLScTs7AMX6wXjqvGbWhz_O4GIxe5qfSIwFxq5FotHVMq7P8UrQ/viewform?usp=sf_link

19 https://drive.google.com/file/d/1fXqwKNxQAxGGUA2viFV7juhWBbbinviD/view

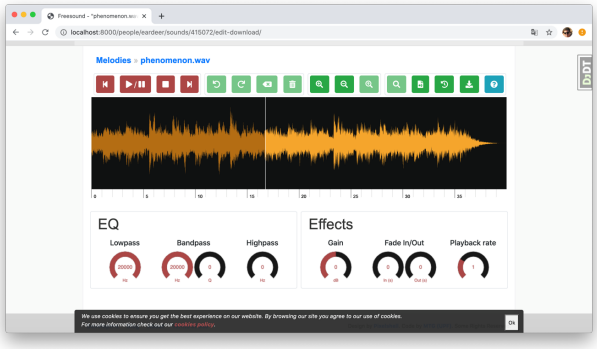

Figure 6. Second version after first evaluation.

As for this second evaluation, the idea was to have a representative group of 5 to 10 people that would run the different user cases in order to test all the functionalities of the editor. However, and due to time and schedule constraints, this approach was not possible, and we finally performed all the user cases on our own. The idea is to perform this evaluation with potential skilled and not skilled users once the editor is deployed in Freesound in order to see if some adjustments are required or not.

For the first iteration of the user cases, the goal was to test if the actual workflow of all the functionalities was the expected one. This was done following the steps of each individual user case and comparing the output of the editor to the expected behavior of the action performed. After this first iteration of the user cases, the workflow for all the functionalities was the expected one, with the exception of the fade in and out effects.

When testing these effects, we noticed that the changes on gain at the start and end of the sample were noticed when playing the sound on the editor but did not reflect in the downloaded sample. In order to solve this problem, an approach using the OfflineAudioEditor was used to render the buffer just after applying the fade, loading the rendered one in the waveform to be sure that the progressive change in gain was applied and saved. Furthermore, the workflow used for the fades was changed in order to match the one used in other editors such as Audacity and similar ones: instead of selecting the duration of the fade, the user will select a region of the sample in which the fade will be applied and then apply the effect.

After the fades were fixed, another iteration of the user cases was held with the same purpose as the first one. Although only the fades workflow was changed, all the user cases must be retested because some functionality might stop working during the development of the new features due to incompatibilities or typos in the code. Finally, all the user cases, including the fades one passed, that is, all the actual workflows of the editor worked as expected and the editor was ready to go.

As explained previously, all the functionalities (actions, filters and effects) are displayed in a single screen, improving the user experience while using the editor (Figure 7). Furthermore, a standalone version of the editor containing all its functionalities was deployed for its testing. It can be found at: https://robertops18.github.io/audio-editor-js/.

5. CONCLUSIONS AND FUTURE WORK

This idea of adding a simple editor to the Freesound webpage will add some huge value to it and will differentiate it from other similar environments and communities, because, to the best of our knowledge, no other similar webpage offers this feature to their users. On top of that, the user's workflow when downloading a sound for their projects will be really simplified, because of the fact that they will only have to use to Freesound webpage in order to edit and download the sample, without having to open and use the editor of their choose. Moreover, the editor has been built in order to let unexperienced users without editing skills be able to use the simpler actions and effects without any previous knowledge about the field. The selection of this type of design over other options like integrating the open source AudioMass project, a fully featured and powerful editor, was made because not every user that uses Freesound knows how to properly use all the audio features that a complete editor offers, so it would be

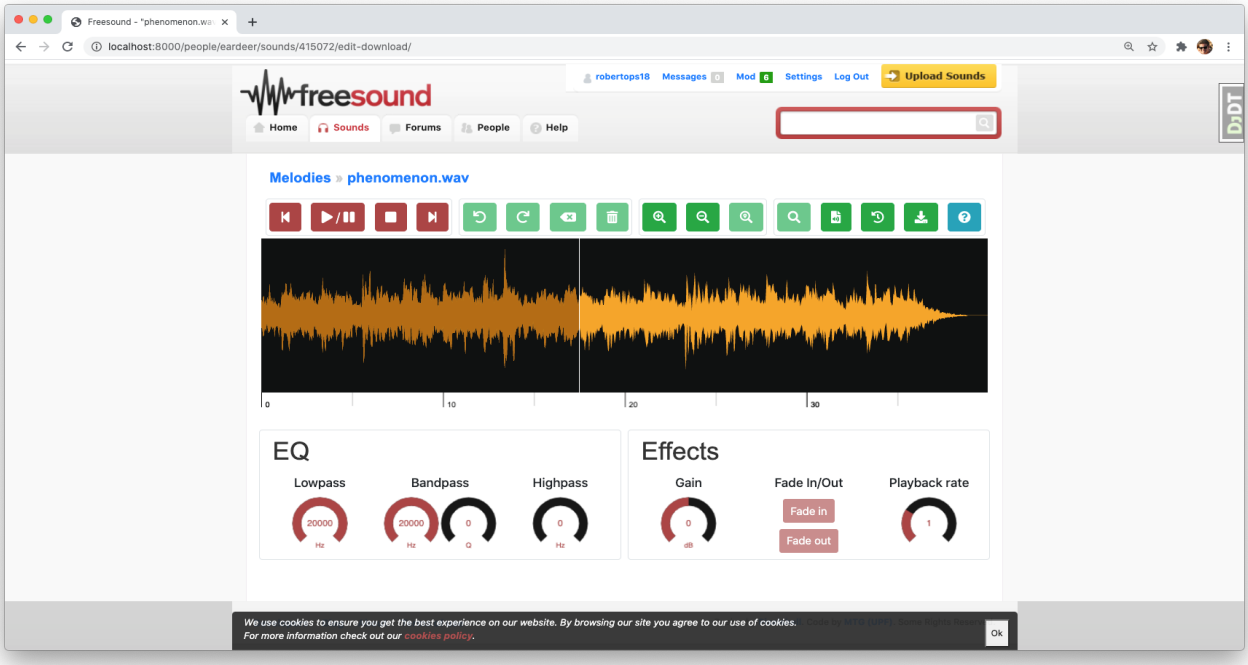

Figure 7. Final version after user cases.

better to have a simpler option where everyone is able to slightly edit the sample prior to its download if needed, but including some more advanced options, like the usage of the available filters in the EQ panel.

In addition, the technologies used for the development of the editor makes it available in all operating systems, including mobiles, and modern available browsers. The code of the project is open-source and available in Github too, so anyone can add or request functionalities that are not currently available in the editor.

Finally, regarding the possible extensions for this editor, more advanced features can be developed like including more gain related effects or more configurable parameters for the filters, like resonance and gain for all of them, as well as adding more filters like the peaking or notch ones. Another great addition will be to be able to adapt the editor in order to let the users edit the sample prior to its upload to the website. This feature will add even more value to the user experience on the website, giving them a full sound editing and downloading online workflow.

6. REFERENCES

[1] Font, F. The Freesound Blog: 2019 in numbers. Retrieved from https://blog.freesound.org/?p=1084

[2] Font, F., Roma G., Herrera P., & Serra X. (2012). Characterization of the Freesound online community. *2012 3rd International Workshop on Cognitive Information Processing, CIP 2012*.

[3] Fonseca, E., Pons, J., Favory, X., Font, F., Bogdanov, D., Ferraro, A., ... Serra, X. (2017). Freesound datasets: A platform for the creation of open audio datasets. *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017*, (October), 486–493.

[4] Smus, B. (2013). *Web Audio API*. Retrieved from http://it-ebooks.info/book/2072/%5Cnpapers3://publication/uuid/F8F580CC-5306-47D0-BEA7-4AFDBD0F5D32

[5] "The Best Audio Editing Software: 11 Audio Editors for Any Situation." [Online]. Available: https://zapier.com/blog/best-audio-editor/

[6] "The 5 Best Free Online Audio Editors on the Web." [Online]. Available: https://www.makeuseof.com/tag/free-online-audio-editors/

[7] B. Dias, M. E. P. Davies, D. M. Matos, and H. S. Pinto, "Time Stretching & Pitch Shifting with the Web Audio API: Where are we at?" Proc. Int. Web Audio Conf., 2016.

[8] A. Milo, S. Rudan, A. Xambó, G. Fazekas, M. Barthet. "Deliverable D6.12: Report on the evaluation of the ACE from a holistic and technological perspective". Available: https://www.audiocommons.org/materials/