# Code Review - Week 2: ToneVision

**Reviewer: Spotlight (Jaime Sanchez, Karan Singhal, Stefan Dimitrov)**

**Reviewee: ToneVision (Jonathan Gilad)**

Met on Friday Feb 20th to discuss project.

## Overview

ToneVision seems to be on a clear track now as opposed to last semester. From what we were able gather from our meeting, they now how have a clear structure for their different components of their GUI programmable processor. They are using a webapp to interface with an arduino chip that will be taking in an input (analog sound) and applying a set of effects to that input signal to produce a desired output. The idea is novel one in that it's trying to modularize and diversify a very specialized piece of hardware in the music industry (muic pedals).

We were shown only what they currently have of their webapp/GUI. They are at a post-preliminary stage in development. They have a working GUI and have started to fiddle with the interface that communicate with the back-end. This review will, then, focus on those two aspects: The front end and what they have so far of the back-end interfacing.

## General

The code works for the stage it is at the moment. The user is able to select a series of modules, link them together and the GUI produces an output that will be sent to the Arduino. The following are some observations that should be paid attention to:

- It was difficult at first to see which files we should focus for the code review. Specify which files are being used. There is a file "script.js" that we spend time reading but later found it was not included in the html page. We recommend there be a better file structure, as there are many "js" and "css" folders and it can become a bit ambigous.
- As this project gets bigger, there will be more files to worry about. Also, more events and models. We recommmend the use of a basic MVC structure or even a front-end framework to keep everything organized better. (i.e. angular backbone, ember)
- One thing to consider is the format of the output. At it's current stage, it separates values of interest with comma and semicolon. We would recommend that they use a more widely accepted fomartting like JSON. We believe this makes it easier to produce (since it is being produced from javascript) and also there are a number of existing tool to parse json that would make it easier on the back-end to employ.
- All the js files are included at the top, the convention is to put them at the end.

- The GUI is very obvious enough when explained but it should include at least a paragraph explaining how to use it.

- There is good modularity in the code. And global variables are used to efficiently identify DOM elements.

- There are superfluous console messages that should be addresed.

## Security

There are some security concerns that come from making a web-app, however, we are aware that this is meant for a more localized approach that a typical web-app, it depends on a third-party Arduino chip. This is our security assessment:

- There is no user input to worry about for now. Since it is all generated based on a predetermined amount of permutations, there should not be a fear of the user tampering with it. If there is any functionality to add user input in the future, make sure it is sanitized it is sent to the arduino.
- In theory, as it is now, the user( by using your js code) can send any number of text to a connected arduino. Keep in mind that how you will control the information flow from the GUI to the arduino.

## Documentation

Overall, there is a clear need for a README that explains what to look at and what is expected to happen when code is run. That being said, there are function contracts for the most important functions, which is very helpful. These are some other (more granular) observations we made in regards to the project's documentation:

- There are a number of code lines that are commented out without explanation.
- It is a convention to document third-party libraries.
- When encountered with incomplete code, ToneVision did a good job in marking it and telling what needs to be done to complete it with a "TODO" marker.

## Testing

Given the scope, it was easy to test the progress that ToneVision has achieved. We simply loaded the file on a browser and tested using the console. Since all the dependecies are included , it was easy to start with the testing without any hassle.