**Jonathan Beuzelin**

Drumset

**Made: July 2018**

# Overview

This Drum Set was designed to create a sound from a designated Key to a Wav file containing the sounds and to include animation on activation using HTML CSS and JavaScript.

For this project you will find the script within the HTML tag as i found it easy to refer back and forth to attach the corresponding keys but this could easily turned into an external file.

I do not know yet how to make the drumset responsive to touch commands such as on a mobile or tablet so this is a project made for desktop.

# Goals

1. **To create a fun interactive way to play the drums:** With this project i tried to take user experience into consideration and how they would enjoy it.
2. **To practice using Javascript and integrate them with HTML and CSS:** The project was my first attempt at combining these three languages to create a interactive user experience.

# Details

**The HTML**

So i began by creating the divs composing of nine different containers which had the same width and height, my intention being to create a clear indication of what a user had to do to interact with the program.

I used the key class to create the containers and then the keys themselves within that container with a kbd tag to create a monospace font to differentiate from the sound name with i put in roboto slab which i felt looked good and was easy to read alongside the light grey colour along with it.

**Javascript**

I then attached the data-key attribute to the the specifics divs for example D had the key code 68 which is its own data code which i found on a website keycode.info and attached them to an audio tag with the src file being the audio in the description of each key so that when the user fires off the key it will find the audio element and then find the key div and then animate it on interaction.

I started by setting up the function playSound with a parameter of e which is short for the event.

I then set up the constant audio which using the DOM and used an attribute selector for it to select the audio element matching with the keycode.

During this process i tested to make sure the keycode was registering properly with the console.log command.

I then implemented a conditional statement to allow for if a keycode was not registered to the audio tag then it would stop the function from running with a return.

I then added audio.play() to see if it would play different sounds on the different key codes within the function and a audio.currentTime = 0 So that the user did not have to wait for the sound to complete and would basically reset the sound if the key was pressed again.

Now that the sound was working i started working on the animation and connecting the key-code to the corresponding key which i did with another const with the name key which would do the same as const audio (listen for the key event and then go for the corresponding key attached to the div.).

It would then apply a .playing css style to it after being pressed which would increase its scale and border size and colour by adding which i did with the command key.classList.add.

Then i had to work on getting the animation to go away as it had added the previous mentioned .playing but there was nothing to take it away so it would just stay in its new form.

I did this by creating a function to remove called removeTransition which would using the DOM querySelectorAll i would look for the .key element and then for each key it would listen for the event transitionend and then run the function.

I used the console.log to make sure the transitionend was being detected just in case i needed to debug.

I then put in a conditional statement if e.propertyName did not equal transform the function would return which would stop the function from removing the classList until the animation had completed.

‘This’ is equal to key so to finalise the function to remove the transition i put in the code to access the classList and then to remove the class “playing”.

For the events to fire off i added an event listener for a keydown event and then the function to go along with that would be playSound which after the transition ended it would then fire off the removeTransition function.

**CSS**

My style is very much flair and i wanted the user to feel they were making a real impact on the page when they were hitting the buttons.

I started with a background that is 100vh and a width of 100% to be a fluid background that would adjust across different monitor sizes with changing font sizes and different widths and padding for the keys.

The transition which is under the class playing is an increase in scale by 1.7, a border of solid silver at 6px, an increase in font size, a color of gold for the text.

I've reduced the margin at the bottom by -50px to reduce the moving of the other keys on the screen and the transition takes .07s.

# What i learnt

The project was my most complex and took me a while of referring to different resources and what was important to me was being able to understand what the programme was doing as i felt just as with any language if i cannot read it then i would not be able to tweak and adjust it.

I had my hand held throughout the course of this project but i hope it attests to my independent working, finding resources to learn from and ambition when it comes to working on these projects.

This was my first instance of using the DOM and i can see why it is so important in JavaScript of being able to select various things around the page and include them into a function.

I had used functions before but not to this degree and at first i found them complex but after learning the flow of the program i found it much easier to read and this has made it easier to write more complex functions than i had before.

This project also allowed me to get to grips better with the console and its relationship with the event listener function while also seeing how it interacts with the functions and has led me to think of other projects such as a calculator or maybe even a board game as some points with a click event.

# What i would improve

I would say overall it's a very beginner project in terms of what it can do where unfortunately i have not added mobile/tablet functionality such as being able to press a key on the screen or maybe even a touchpad desktop monitor to truly make it feel more like a drum set but this is something i plan on adding in the future.

I had to use key up for this project due to if the key was held down the animation would bug and would not be removed but i am still looking for a solution as i feel a key down event would feel better for a user.

If i were to redo the project i would go more in depth into the transition and add more animation to it with different class lists being added for the different keys to give the user a different experience of hitting a different key every time such as a different colour font or maybe the colour of the padding would change colour with a different type of border.

For me atleast i feel the code is readable, which as a beginner it's important for me to be able to understand the code and i feel i have adhered to not repeating myself but maybe i could look at combining the functions so it's all just one function to compress it while keeping its readability.

Thank you for reading,

Jonathan Beuzelin