CS2003 Practical 2 report

Tutor: Xu Zhu

Matriculation ID: 210020020

Overview

This practical was to take a csv file of every Beatles song (that had an album release) and convert it to json using node.js, then create a website with html, css and js to create a playlist using of Beatles songs.

Overall I have completed the entire specification plus some small extensions of my own creation and would classify this practical as a success.

Design

As this practical has multiple components I will split the design section into multiple parts for each: the Node.js converter, the main javascript component, and the html and css parts.

Also note that the implementation was entirely built upon html5 boilerplate[1], it doesn't affect my actual design or implementation, its main use was for place holder icons as well as normalisation for css.

Node.js:

Fairly simple as it was mainly based on an example already given to us in lectures and on studres[2]. Not much needs to be said on this other than removing the sorting as that was unnecessary, changing the formatting of split and the variables to the csv file given to us, and changing the output to a file instead of stout. I also changed the fs.exists function used as it was depreciated.

I was considering taking in command line arguments for the file path, however I decided against it as its not very necessary for this practical for the converter to be modular as the main js file already has the file input for json hardcoded and its safe to assume that file would always exist.

HTML & CSS:

Figure 1 below is the rough wireframe for the html I followed for the website, it uses two grid layouts, one to split the header, playlist and song list, and another for each song to split the name, album, year, duration, and album cover (note: the songs are not in the exact format shown in the

specification, however I believe it still gets all the information across, looks nicer and does not require substantially more or less work.)

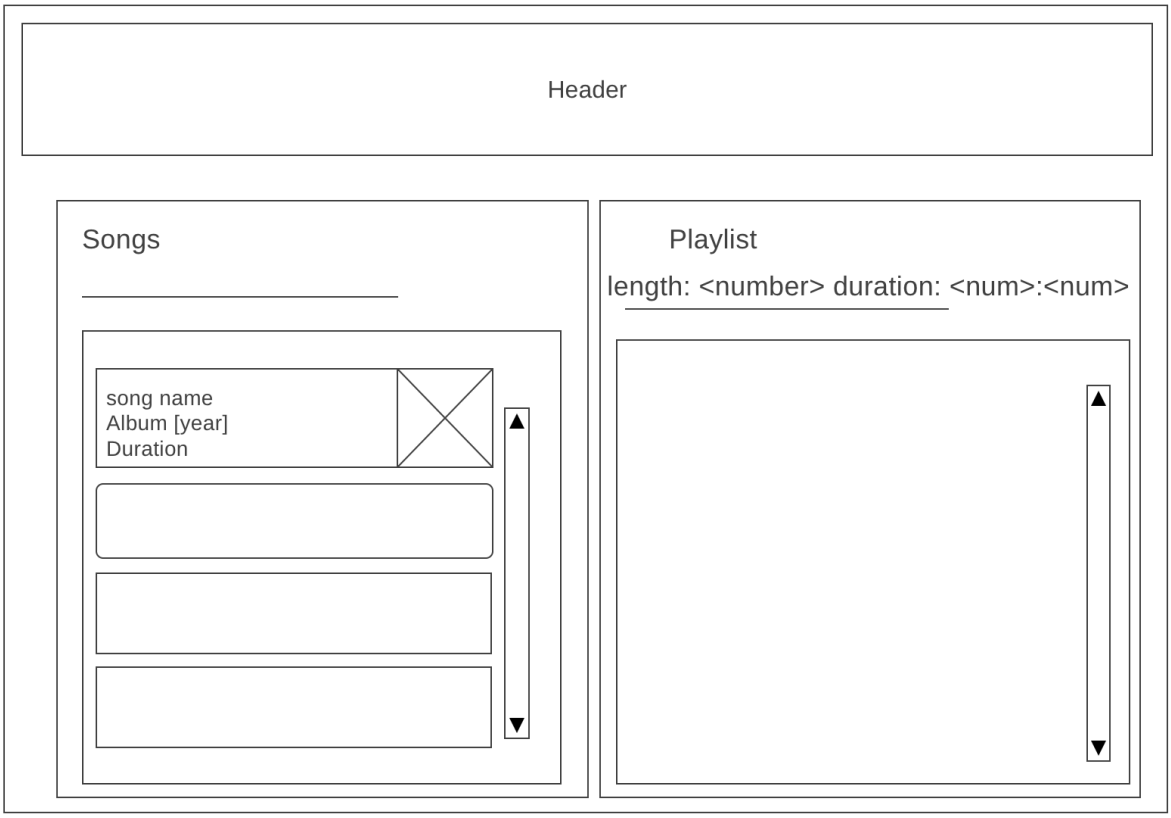

Figure 1: html wireframe

I also used a custom font in the style of Sgt peppers lonely hearts club's album cover[3] it is licensed as non commercial freeware and should be fine to use for the purposes of this practical.

I also consulted Özgür on whether we would be allowed to use the album covers of the Beatles within the practical, he did allow this but urged me to check that this falls within proper use in copyright law first. After consulting some documents provided by the UK governments Intellectual property Office[4][5] I found that as my use of the album covers wouldn't affect sales of the original product nor work as a substitute for how it was intended to be used, the usage would likely come under fair use. However to be on the safe side I consulted a friend and artist Vlad Tronciu (vt29@st-a) to commission them to create parody drawings of the album covers using Microsoft Paint, as in UK law parody drawings are also a defence in fair use, as is sufficient

acknowledgment, however the original artist of the album covers, Richard Hamilton, died in 2011 and it is difficult to find who the correct copyright owner of the album covers are currently, especially as the Beatles have famously had a long history of battles on who owns the copyright on their catalogue of songs. Vlad has given me permission to use his parody drawings for this practical where he has completed them, and kindly given me the works in progress where he hasn't completed them in order to finish them myself, as you can clearly see from the differences in some of the album covers, our artistic styles are practically indistinguishable. (See attached figures 2 and 3 at the bottom of this report.)

The css was mainly to create the two grid templates that the website is based on. I decided to use the Grid layout as its incredibly powerful and more versatile than flex boxes due to being two dimensional thus not requiring nested flex boxes for the grid layout used in each song entry in the list

Css was also used to solve some peculiar edge case behaviour I have found, for instance the background colour did not reach the bottom of the screen in either chrome nor Firefox, I fixed this by setting the min height of html and body to 100%, although I am not entirely sure why this works as I would assume the body of a page would always be 100% of said page?

Also whilst Özgür told us in a lecture that this practical won't be marked on responsiveness I did do some mild testing using the Firefox dev tools and found that the website requires a screen of size 700x500 in order to not have any overflow, it should work as expected otherwise.

JS:

I used the fetch api to get the file for the website as it's much easier to use than the other suggested ways to get the file in a robust way, as it has its own catch syntax via promises.

For each song within the json file I used tagged templates[6] in order to write the custom html in a uniform way, I also considered using custom html elements extending list in order to implement something similar, however this way seemed much simpler and easier to read to me.

I also originally had a button to move the songs on and off the playlist, however I replaced this by attaching 3 event listeners to each list item, one to do the moving back and forth, and two onmouseover/off to change the colour of the card the user is on to show it can be moved, I believe this is as obvious as a button and looks nicer as it gives a more responsive feel to the website.

When moving the song to the playlist, rather than removing it, the list item is just set to hidden, this way the song list will always be in the original order, just without the songs in the playlist.

I also didn't allow duplicates in the playlist as I feel that's not in the spirit of how a playlist is usually set up, I may have changed my mind as I was implementing the ability to sort the playlist by dragging and dropping the songs, however this functionality had to be scrapped due to time constraints.

Testing

All tests have been run on both Firefox and chrome where applicable.

What is being tested	Pre-conditions	Expected Outcome	Actual Outcome
Csv converter with no file	Running csv converter with the beatles cleaned csv file removed	../TheBeatlesCleaned.csv: no such file	../TheBeatlesCleaned.csv: no such file
All songs are correctly loaded into website	Running console.log(document.getElementById("songList").children.length) on a freshly loaded website	193	193
That adding to playlist works correctly and total length works as expected	Adding all songs to the playlist on a fresh reload and checking console.log(document.getElementById("songList").children.length) is equal to totalLength	193 193 true	193 193 true See figure 4
That removing from playlist works	Taking one song away from the playlist outputs the correct number	192	192

What is being tested	Pre-conditions	Expected Outcome	Actual Outcome
That the scroll bar for both songs and playlist work fine	Adding the 14 songs from please please me should have both songs and playlist have separate scrollbars and no overflow	TRUE	See figure 5
Adding a song updates the duration and song count as expected	Adding all 14 songs from please please me should give the correct total length and duration	14, 32:31	14, 32:31 see figure 5
Removing a song updates the duration and song count as expected	Removing misery from the above test	13, 30:42	See figure 6
Songs are added to the playlist in the order added	Misery was the last added despite being track 2 so should appear last	TRUE	See figure 5
Songs removed are returned to their original position	Misery should return to the top of songs once removed	TRUE	See figure 6

Evaluation/Conclusion

In conclusion I would say this practical has been a complete success, I have implemented the entire specification, as well as some features that go beyond the spec.

Most difficulties I had in this practical didn't particularly come up in the design, but I did have a very hard time getting my head around javascript as

its my first time properly working with the language, and at the end of this practical I can actually use it to roughly the same degree I can java I would say, but it was a very steep learning curve for me, particularly for things not covered in lectures till later in this practical that I had to use early on, such as working with promises and async functions. My initial use of the fetch api involved 2 async functions with one working as a wrapper as I wasn't familiar with the correct way to use promises yet.

Given more time I really would have liked to implement the sorting by dragging and dropping, I just wasn't quite able to figure out how to organise a list with non standard list elements on time, I also would have liked to investigate implementing the Spotify api to see if it would have been at all possible to export the playlist to third parties.

Bibliography

1. https://html5boilerplate.com/

2. https://studres.cs.st-andrews.ac.uk/CS2003/Examples/web/node/web2/node-01/09-csv-files.js

3. https://www.fontspace.com/sgt-peppers-outline-font-f23684

4. https://www.gov.uk/guidance/exceptions-to-copyright

5. https://www.gov.uk/government/publications/changes-to-copyright-law

6. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals#tagged_templates

Figures:

2. an album cover of "beatles for sale" by Vlad

3. An album cover of “let it be” by me

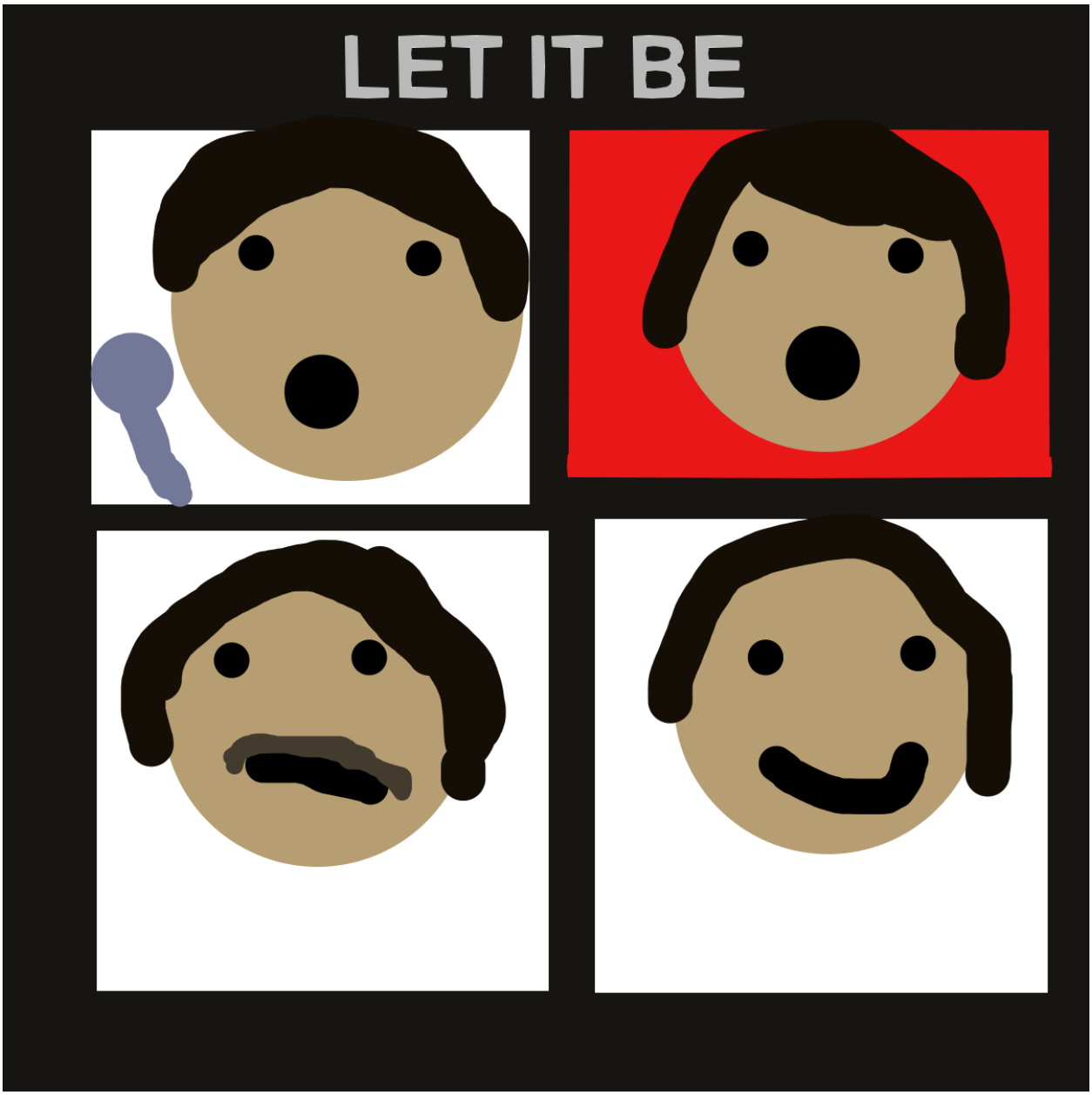

Figure 4:

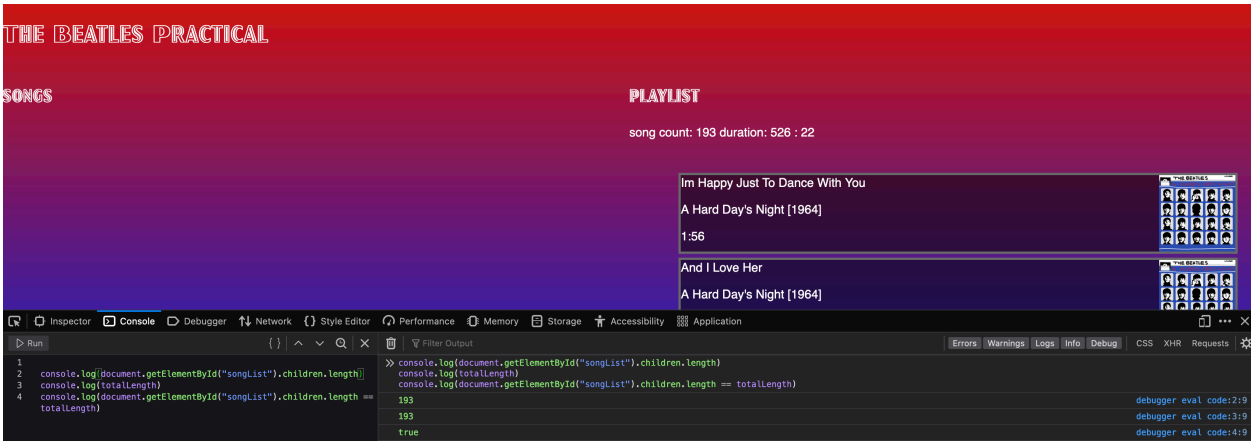

Figure 5:

Figure 6:

THE BEATLES PRACTICAL

SONGS

Misery
Please Please Me [1963]
1:49

It Wont Be Long
With The Beatles [1963]
2:13

All I've Got To Do
With The Beatles [1963]
2:03

All My Loving
With The Beatles [1963]
2:08

Don't Bother Me
With The Beatles [1963]
2:28

Little Child
With The Beatles [1963]
1:46

Till There Was You
With The Beatles [1963]

PLAYLIST

song count: 13 duration: 30 : 42

Please Please Me [1963]
2:22

P.S. I Love You
Please Please Me [1963]
2:04

Baby It's You
Please Please Me [1963]
2:41

Do You Want to Know a Secret
Please Please Me [1963]
1:57

Taste of Honey
Please Please Me [1963]
2:03

There's a Place
Please Please Me [1963]
1:50

Twist and Shout
Please Please Me [1963]
2:35