Learning diary: Niko Kareno
Anytime-course: Software Development Skills: Front-End 2020-21

Introduction to workflow and sass:

I've had some previous experience working with VSCode and Node, so setting up the project was quite familiar. The lesson introduced some nice new 'features' to my workflow which I will definitely be using in future (Bracket pair colorizer and Emmet tips for example). I'm not a very confident user of Git, so it was also a nice thing to go through the basic steps and actually commit the project along the way. Repeating the init steps helps to remember what's happening in git.

At this point I'm still slightly confused about the terminology when it comes to SCSS and SASS, but I hope this will clear out a bit more during the course. I've had some experience with CSS and vaguely remember that seeing the changes on the browser seemed to be a bit tricky in some cases (I guess some compiling issue?). So far the sass script seems to work nicely in this case.

Homepage and Core Sass/CSS:

This part of the course was somewhat familiar to me already. Creating the basic HTML and CSS for the page was of course a good recap on CSS syntax and how things like padding and margins behave. I've been mainly using % and px units, so rem and vh were a new acquaintance to me.

In the nesting of the CSS I got to know some new concepts I was not familiar with before - which made me rethink some of the CSS structures I've been creating before. I find it interesting and a bit frustrating at the same time that the languages / frameworks are quite dynamic and features that are added later on can have a great impact on the structure of the code. I guess refactoring is something that is continuous work, no matter how well you structure your code in the beginning - at some point there will be new features that push for the need for refactoring.

Rotating Menu Button

Making the rotating menu button was interesting. From the final result it's actually quite difficult to realize that the 'x' is formed from the lines of the button, which is pretty cool in my opinion. Also the concept of having the whole menu inside the index page is a bit controversial for me. I've had some experience with angular framework and not having the menu as a separate component / page seems a bit wild. I'm still learning along the way so seeing different types of projects is interesting - in this particular case the structure of the page seems very simple and the use case is quite clear, so I guess this kind of structure serves the purpose. On the other hand I feel like it would make sense in wrapping these nice and tidy features into components that could be easier to use in other projects - but I do realize that it's a completely different topic.

Menu Overlay & Responsiveness

Somewhat continuing on the previous lesson (I did both of these at once): Having the menu overlay as a pure CSS feature was very interesting. As said before, it was a bit new idea for me to be able to create such a nice menu without having it as a completely separate component - although the scss file structure sort of makes it a component. Understanding all the different possibilities for structuring the code / project is a bit overwhelming at the moment.

Page With CSS Grid

Use of the grid was new to me. I have some experience with html / css from years ago and back then tables were the way to go. The grid layout seems to be TONS better. Also by attaching the media queries to the screen size made the responsiveness feel very nice.

Work and Contact Pages

The grid and flexbox both seem to have their own advantages. I'll be looking into them more on my own project. Even though they make the layout handling a lot nicer, on the first touch I was having a lot of trouble trying to get the exact layout I wanted by tweaking the properties. It seems that although they are a massive improvement to some of the old ways - they are still not super intuitive (for me at least).

Website Deployment

Doing the actual deployment was surprisingly easy. Also the node script for running the deployment from the bash console made it nice and easy to do deployments. Both the css rebuilding and deployment scripts were something completely new to me and nice to get to know these.

Own Project

I tumbled down the rabbit hole with the grid and flex layouts in my own project. My goal was to make a grid / flex testing editor where one can test how the layouts behave with different settings. Getting the basic settings working was quite ok, but once I started going deeper, I ran across some issues that I was not able to solve (or would have taken too much time).

I wanted to make it possible for the user to change the grid box size both column and row wise by using the “grid-column-start” and “grid-column-end” properties. The problem I encountered was that when the grid is drawn on “auto” settings and when for example the layout gets changed - there is no proper way to get the coordinates of the grid items from ‘getComputedStyles()’ in JS. So the only way (I thought) would be to write the code for calculating all the coordinates of the boxes using the grid-template-columns.lenght and grid-template-rows.length and matching them with the relative positions of the elements in the grid. This got way too complicated for me, so I decided to leave this out.

For the menu overlays, I practiced a bit by changing the overlay directions and transformations. I did not spend too much time on this, since I was so excited about my grid experiments.