Styling your Submenu

This lesson briefly explains how to approach styling and how to implement them via code.

WE'LL COVER THE FOLLOWING

- Placement of submenu items
- Finishing off submenu styling

Placement of submenu items

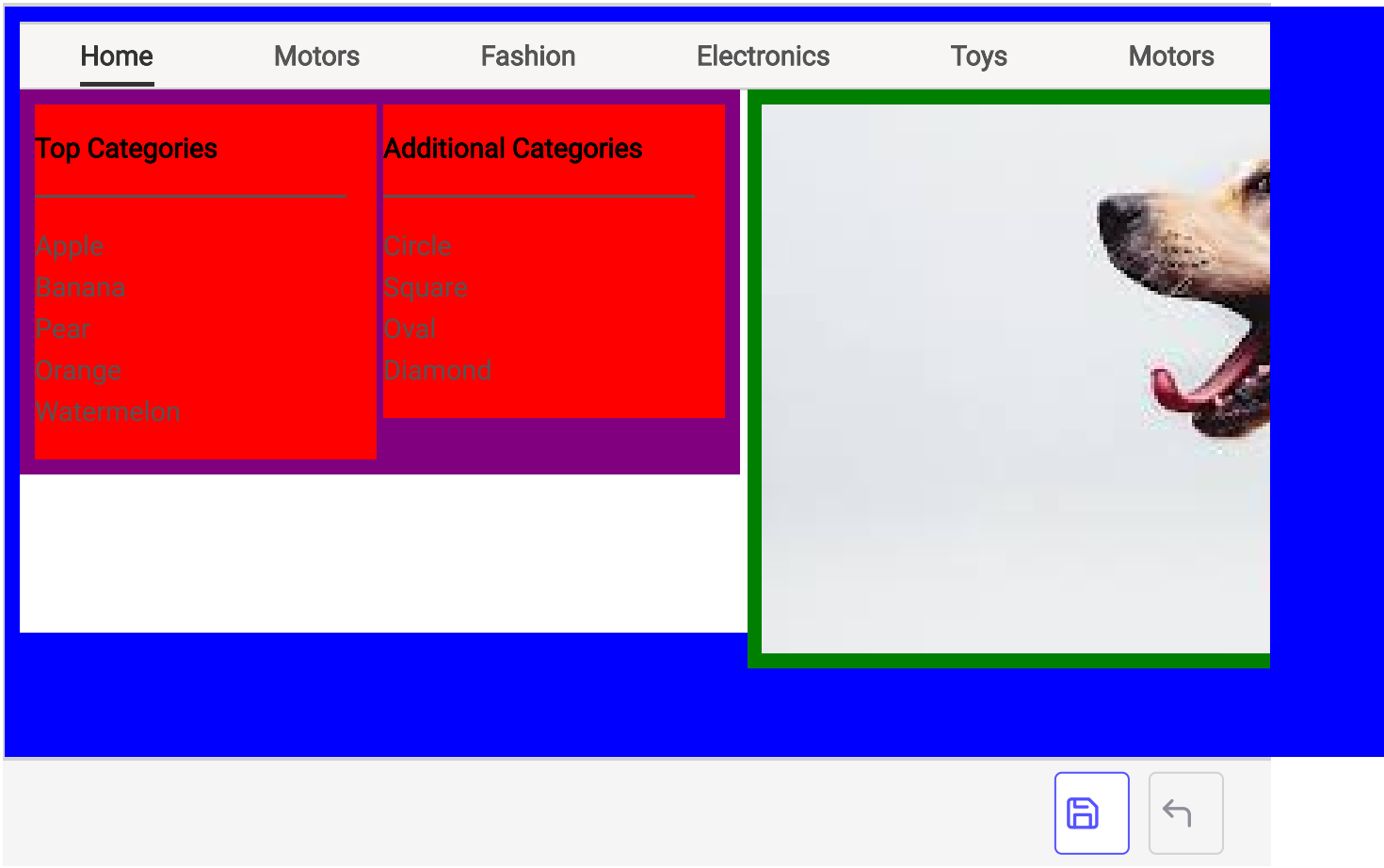
Just like with the horizontal orientation of the top-level menu, we'll use `display: inline-block` on the elements that need to line up horizontally with each other. It looks like we can use percentages for the wrapper `div`s that delineate text from the image – something like 40%/60%, and 50%/50% for the two inner columns. We need to size the text correctly, align it all to the left, and remove the default bullet point configuration of `ul`. Lastly, we can add the border to the header text and add paddings to everything.

By the way, the technique of excessively using high-contrast background colors to make elements clearer while aligning applies for inner `div`s just as much as the outer.

Output

HTML

CSS (SCSS)



In addition to the changes we discussed, there's a few more that are worth noting:

- Define variables for shared colors
  - I recognized that some of the border and text colors were the same as the ones used in the main menu. Instead of repeating the colors, I defined variables and referenced those in the places where colors were shared. Not repeating yourself is good practice even in CSS (though it's often unavoidable)!
  - How you define variables depends on which preprocessor you're using, if any. The syntax I've used is for native CSS (nothing additional required).
- I had to add `vertical-align: top` to elements. This says that when there are two elements in the same row with heights that don't match, we should align them according to the value in this property. Without this specification, the columns would've lined up like so:

- I realized that the only way to have enough room for two text columns of a reasonable font is to expand the menu like the example has it. So I just copy and pasted the elements in my HTML.
- We globally added the property `box-sizing: border-box`. This means that elements specified dimensions exclude padding and borders. It's included so we can do calculations correctly. For more information, see https://css-tricks.com/box-sizing/
- To set the widths of our newly created elements, we use `calc`. This lets us mix different units, which is useful when we have a ratio in mind but need to use other units. In this case, we're subtracting some pixels to make the elements line up properly in the same row.

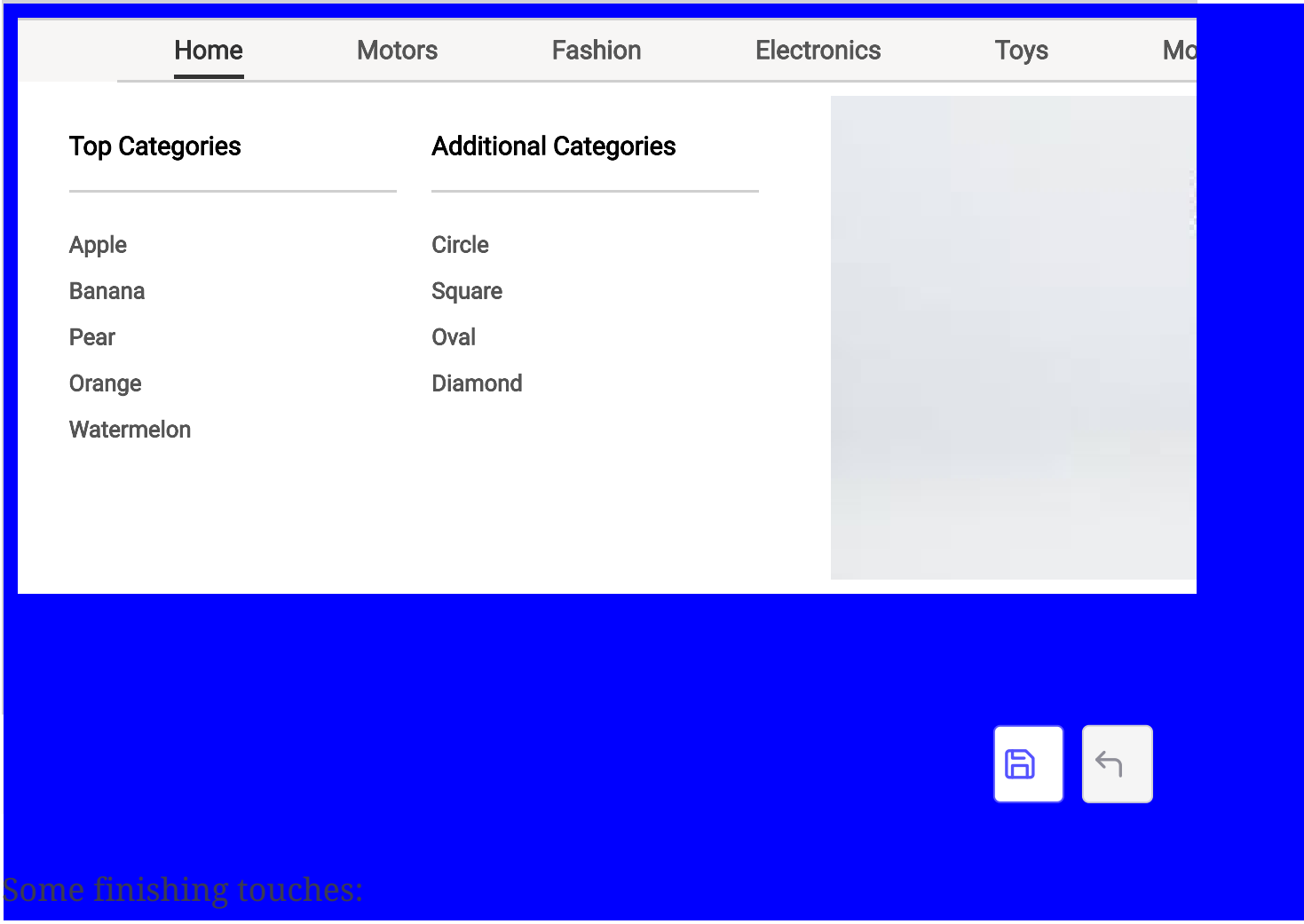
We can now start removing the high contrast backgrounds and start making things more final.

Finishing off submenu styling

Output

HTML

CSS (SCSS)



Some finishing touches:

- I noticed the menu items don't actually extend all the way to the end of the menu, so I gave the `menu` a bit of extra width and set `width:90%` and centered it on `menu__main`. Even though it was previously unused, well-planned groupings make it easy to customize in the future!
- To have the image contained within the `div`, I set both width and height to 100%.
- To better target the links, I realized I should've wrapped the actual text of the items in each category around a `<a>` tag since they are links to another page.

I think this looks pretty close to what we want the outcome to look like. In the next lesson, we'll add the functionality in order to have different items appear, depending on the item that we hover over.