COMP6080: Web Front-End Programming

Tutorial 5

Sidney Pham
sidney.pham@unsw.edu.au

UNSW

Week 5, Term 3 2020

Some Unpleasant Code

Look at `review.html`. What is wrong with the style of the source code for this page? What changes would you make?

A: Loops and functions would reduce code repetition.

Debugging

Why won't this page work correctly?

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <div id="dogname"></div>
    <script type="text/javascript">
      const data = fetch(
"http://www.cse.unsw.edu.au/~cs6080/20T3/data/doggo.txt"
      );
      document.getElementById("dogname").innerText = data;
    </script>
  </body>
</html>
```

`fetch` Returns a Promise!

You'll want to do something like:

```
fetch(URL)
  .then((res) => res.text())
  .then((data) => {
    document.getElementById("dogname").innerText = data;
  });
```

Fetch `Response` objects have methods like `.json` and `.text` that return a promise that resolves with the value you probably want.

Site Study

What are some accessibility issues with `https://mcdonalds.com.au/` (the site and its source code)?

One thing I can spot is poor text contrast! The W3C's Web Content Accessibility Guidelines (WCAG) tell us about contrast ratio. One useful site is `https://contrast-ratio.com`. Another is Google's Designing in the Browser Youtube series.

Lighthouse

Chrome has in its developer tools an audit tool called Lighthouse, and it tells us plenty of stuff like:

- Buttons do not have accessible names
- Image elements do have `alt` attributes
- Form elements do not have associated `labels`
- Links do not have a discernible `name` attribute
- HTML element does not have a lang attribute

You can run this on your *own* site!

Exercise: A Rough SPA

Build a page in `tabs.html` that has:

- *A header bar with options Home, About, News, Articles*
- *A footer with a contact us link*

When the items in the header bar are clicked, the body of the page should change to the relevant content below.

NOTE: You aren't to just change the inner HTML of the body element, as there may be many nested HTML elements within that. You can either solve this via rendering all bodies and toggling which is displayed, or constantly re-rendering whichever one is being displayed currently.

Warnings

Be careful with SPAs if you want nice URLs and history. Clicking on a link doesn't do anything to the URL. You also can't use the back and forwards buttons like with static sites! Tools like React Router handle this by using the History API.

(Non-Assessable) Beware: `setTimeout`

`setTimeout` is great, and very often useful, but when you need to be a bit more precise about time, you should be careful.

For example, I had to make a timer app recently, but I decremented by one second each time the timer fired. You learnt this in your lectures too, but as a reminder: timers can be somewhat inconsistent (especially when the window isn't focused!). You should generally use performance.now to find out how much time has elapsed.

Monotonic vs Non-Monotonic Time

Generally, in JavaScript, you should use `performance.now` (as opposed to `Date.now`) to find out how much time has elapsed, becuase it gives you monotonic time. This means it will only ever count upwards. This is generally something you want in a timer.

But be careful, `performance.now` might not tick when the user puts their computer to sleep, which is a strong argument in favor of `Date.now`.

The Timer App

If you were curious, I'll also share the timer app code, because there are a few cool techniques like using `requestAnimationFrame` to render declaratively and a simplified tagged union.