Lucas Kitaev

Mr. Combs

Senior Projects

10 November 2017

Journal 1

Goals

- Finalize website style
- Add bibliography, goals, and journals pages to the website
- Finalize website script
- Register custom domain for website
- Complete Java lessons on exceptions and iterators
- Complete Java lesson on sets
- Complete Java lesson on maps

Research

These weeks were mainly focused on improving my project website to look and act somewhat professional, although near the end I began to explore the AP Computer Science AB curriculum. To accomplish this, the brunt of my research was focused on refreshing and expanding my knowledge of Cascading Style Sheets (CSS) and JavaScript (JS).

Cascading Style Sheets. Used as the main language used for styling websites, it is imperative for web developers to become comfortable with CSS. The language is structured using selectors and

declarations. Selectors can target elements of a single type, like `img`; specified element classes, like `.box`; unique element ID's, like `#tip`; pseudo-classes that represent a certain state of an element, like `:hover`; pseudo-elements that represent a generic relationship to another element, like `::after`; or elements with a certain attribute, like `[src]` ("CSS Selector Reference"). As I recall from my previous CSS experience, declarations take the form of a property, like `display`, and a value, like `block`, separated by a colon. Declarations are put into blocks under a selector, with each declaration terminated by a semicolon. The name "cascading" comes from the fact that styles cascade, or overwrite each other. One way that this is accomplished is based on location: a `ul` block placed after another `ul` block will cascade over the first, but only for shared properties (styles don't get thrown away). Styles also cascade based on specificity: a `ul.box` element is more specific than a `ul` element, and as such, the `ul.box` styles will cascade over `ul` styles. Another important feature of CSS I learned is media queries. A media query is its own block, defined by `@media`, with one or more conditions, such as `(max-width: 768px)`, or `screen`, separated by `and` ("CSS3 Media Queries"). I used media queries to target tablets and phones, and applied specific styling to improve the appearance on those smaller screen sizes. CSS is also capable of animating elements, which is a topic I would like to cover in future research.

JavaScript. Similar to CSS, JS is the main language used for the functionality of websites, and is equally important for web developers to understand. JS is capable of controlling nearly anything on a webpage, but due to security reasons, it cannot manipulate embedded content from other websites. JavaScript functionality can be extended using any number of libraries and frameworks; however, for now, I'm focusing on vanilla JavaScript, so I can get used to the

basics. The main piece of information I had to learn about was JavaScript events. Events are triggered by specific actions that happen on the web page, and once an event is triggered, it performs any functions specified in the code ("Introduction to Events"). On my website, I used the `onclick` event to handle the selection and deselection of list elements in my timeline. I also used the `onload` event to set the class of links as either internal (on the same domain) or external (not on the same domain), which allowed them to be styled properly without having to specify the class for all links. The function also sets the appropriate target attribute for external links, to make them open up in a new tab. JS is capable of much more, and if I learn advanced topics as a part of future research, it could be very powerful for client-side functionality, in conjunction with Java (actually unrelated to JavaScript), which provides server-side functionality.

Java. At the time of the writing this journal, I've only completed research on exceptions and iterators, which is mostly review for me. Exceptions are, put simply, errors in the code that arise because of improper logic. If the programmer knows an exception could possibly occur, it is handled by `try, catch,` and `finally`. The `try` block includes the code that may result in an exception, such as `int z = x / y;` where `y` could be zero because of user input, which throws an `ArithmeticException` because the code attempts to divide by zero. This is handled by `catch(ArithmeticException e)` with accompanying code which notifies the user that he attempted bad math. The `finally` block specifies code that should be run regardless of whether the `try` fails or not. Iterators are objects which implement the `Iterable` interface (provides the layout of what classes which implement it must handle). An iterator is simply something that can be traversed; iterators in Java provide methods for accessing

the next and previous elements, as well as adding, removing, and setting elements. The methods `add(E e)`, `set(E e)`, and `remove()` affect the last accessed index by the `next()` or `previous()` methods.[1] Going into the next week, I'll cover more advanced Java concepts, including sets, maps, stacks, and queues.

Accomplishments

- Working JavaScript and CSS[2]
- Understanding of exceptions and iterators in Java

Reflection

I've ended up spending more time working on my project website than I originally planned for. However, this is not problematic, as the web development knowledge gained from spending time on the project website can be used later on when I'm designing the web application itself. My first five goals have been completed; I have a stylesheet, script, and custom domain, as well as all of the necessary pages set up for my website; I've completed the first unit of the AP Computer Science AB curriculum. This still puts me roughly two days behind on my timeline, but at the very least, I'm confident nothing major will have to be pushed back.

[1] E is a generic name for any object in Java. It is usually included in an iterator declaration using the diamond operator. For example, `List<Integer>`.
[2] script.js and style.css

Works Cited

"CSS Selector Reference." *W3Schools*, W3C, www.w3schools.com/cssref/css_selectors.asp.

"CSS3 Media Queries." *W3Schools*, W3C, www.w3schools.com/css/css3_mediaqueries.asp.

"Introduction to Events." *Mozilla Developer Network*, Mozilla, 13 Oct. 2017, developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events.

Jang, Peter. "Modern JavaScript Explained For Dinosaurs – Peter Jang – Medium." *Medium*, Medium, 18 Oct. 2017, medium.com/@peterxjang/modern-javascript-explained-for-dinosaurs-f695e9747b70.