CSSE 290 JavaScript search highlighter in-class exercise

The PHP code that I provide creates a form that allows a user to select a file from a small collection of text files. When the form is submitted, it sends back the contents of the selected file. Each line of the file becomes a paragraph in the HTML file that the server provides to the browser. Once the page has loaded, the JavaScript code that you are to write will set things up so that when the user clicks on any word in the text, all occurrences of that word (even those that include punctuation or a different upper/lowercase configuration) are highlighted.

Unlike other in-class exercises, this one is required. It should be good practice with JavaScript for next Friday's exam.

You and your in-class partner should do it as a pair-programming exercise. In a couple of cases, I will add "partnerless" students as a third person in a group. There should be adequate time to work on it in Monday's and Thursday's classes. You should not have to work on it outside of class time.

Implementation notes.

- You and your partner should pair program on a single computer. I recommend having the partner who feels least confident about JavaScript programming be the driver.
- Note that only "whole words" get highlighted. You do not have to search for the clicked word as a substring of some larger word (except that there may be punctuation).
- I am hoping that you will look through Chapter 9 (or do internet searches) to find the appropriate DOM concepts and methods to use.
- You probably do not need to change anything except the `highlight.js` file.
- You will probably want to write several small functions to handle various parts of this task. For example, for my solution, I wrote five functions. The longest is 12 lines long if I do not count blank lines or lines that are only comments or closing braces.
- With the same kind of counting, my entire solution is 51 lines of code. Add in comments, blank lines, and lines with just }, and I have 74 lines.
- One possible approach is to put each word in its own HTML span. I found it easiest to do that in a copy of the original div, and then eliminate the original div.
- As suggested in section 9.2.3, it is better to put style info in a CSS file (I did this for you) and let the JavaScript code change the class of elements whose styles should change dynamically.
- When I demo my solution in class, you will notice that when the file is large, it takes a few seconds for the page to load, But then when I click on a word, the highlighting happens instantly. This is because as I process the words to put each in its own span, I also build up a dictionary that keeps track of each unique word and all of the spans that contain that word.
- An object (dictionary) with no keys is denoted by { } , and `object.property` is essentially the same as `object["property"]`. One difference is that the latter can more easily be done dynamically at runtime.
- The names of some methods that I used include createElement, createTextNode, appendChild, charAt, getElementsByTagname, removeChild, trim, split, push, match, toLowerCase, remove, add.
- The names of some properties that I used include classlist, length, onload, onclick, parentNode, innerText, id.