Week 4 Research

By: Patrick Corcoran

The difference between var, let and const. Before the JavaScript ES6 var was king of declaring a variable. Var is the only of the 3 that has a global scope, meaning it can be declared outside of a function and used anywhere in the program, but it is able to be locally scoped in a function as well. Var is mutable so I can have its value updated, but also unique to var it can be redeclared later in a program. This can potentially lead to unintended issues where you were expecting a particular value, but get a different one due to redeclaring it later on.

Let and const are both locally scoped variables. If you declare them within a block of code, such as a function they only really exist within that function. Let is mutable like var, but cannot be redeclared. The combination of the local scope and mutability make it fantastic for use in loops. You can declare an iteration variable in a for loop, have it do the work, then be perfectly usable later on in different loops to redo that job. This allows you to use less variables overall instead of having to create a new iterator for each loop.

Const is different in that it is not mutable. Unlike var and let which can be declared and not initialized resulting in an undefined value, const must be initialized. Once declared and initialized it will retain that value permanently. The one exception to this is declaring objects with const, while the object itself cannot be updated, the individual elements within the object can be.

All 3 types are subject to hoisting but each a little differently. Hoisting brings all the variable definitions to the top of the code and declares them regardless of their physical position in your code. Var is both declared and initialized with undefined unless you gave it a value due to its global scope. Let and const are both declared but not initialized due to their local scope.

Some of the new features that were introduced in JavaScript ES6. ES6 released in June of 2015 and included a number of quality-of-life updates to JavaScript's syntax. As discussed in the previous section of this report ES6 introduced let and const for better variable handling.

Template literal was introduced for the output of strings. Instead of having to use the clunky + concatenation methods, wrapping your text in back tics returns literally what is typed, including multiple lines and variables wrapped as such: ${variableName}. Your code appears a lot cleaner with template literal and it's much easier to handle white space without needing special characters.

Arrow functions were also introduced with ES6. Arrow functions allow you to declare and include the logic of simple functions in a single line. Massively increasing the simplicity and readability of your code. They can handle more complex functions as well by wrapping your logic in curly braces, but you do have to explicitly include a return statement once they go beyond a single line of code.

These features along with a lot of other higher-level ones that I don't currently understand were part of a huge update to JavaScript. I am very excited to see how things like promises, classes and modules get integrated into my coding journey in the coming weeks.

References:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements

https://www.freecodecamp.org/news/var-let-and-const-whats-the-difference/

https://medium.com/@kavisha.talsania/top-10-es6-features-every-javascript-developer-must-know-4c81ec54bbcd

https://www.freecodecamp.org/news/these-are-the-features-in-es6-that-you-should-know-1411194c71cb/