Week 04 Research: 04/27/23 Jose G Espino Espino

Prompt 1. What are the differences between var, let, and const?

Var, let, and const are the only forms of declaration in JavaScript. A declaration happens when one of these three keywords comes before a variable. After the declaration, depending on which keyword was used will define the scope of the variable. For variables declared with var, the scope is global, meaning you can access the variable anywhere in your code. For example if you declare a variable with var inside a block of code, that variable can be access outside of the block of code. For variables declared with let or const, you cannot access them outside of the block of code that they were declared in. However, if you declared variables with let or const outside of functions, or outside of blocks of code, they can be used as global variables. The difference being that if the variable is declared inside a block of code, it cannot be accessed outside of that block of code. For variables declared with const, the variable will be protected from re-assignment. This means if a variable is declared as a const, it will permanently have the value that itt was initialized with for the remainder of the code. Declaring variables using this three keywords is key to having scope flow in your code.

Sources: Grammar and types

Prompt 3. What are some features that are new with ES6?

ES6 introduced a lot of new features to JavaScript including Classes, Arrow Functions, and template literals, to name a few. Classes are basically useful when you need to have a group of objects with the same keys, but different values. Once you have declared a class with set keys, you can declared an instance of the class when you declare a variable with the value of 'new myClass()'. This will create a new instance of the myClass object without having to create a completely new object. Another feature introduced in ES6 is arrow functions. They are always unnamed, they replace anonymous functions. The can be written omitting the word function, parentheses, and the return keyword. Not every time, only if the function has one statement in it's body can you omit the return and only if it has one parameter can you omit the parentheses. For example, (a) => a + 100; can be ran without error. Furthermore, if you need to call the function, you can assign a variable to the arrow function. An example would be, let myArrow = (parameter) => parameter + 100;. Lastly, template literals were also introduced in ES6. Their purpose is to have a string without the need for quotes. This way you are able to have functions inside of the string and use variables more freely without having to use escape characters. In my opinion I think it's one of the most useful things that I actually use. In conclusion, ES6 introduced a lot of useful features, most of them were things that previous versions needed and or used a lot of methods but with these new features made it easier to use things that were used very often but now configured as a built-in keywords or methods.

sources: Template Literals

sources: Arrow Functions

sources: Classes