**VARIABLES**

**Create a Variable: var**

There were a lot of changes introduced in the ES6 version of JavaScript in 2015. One of the biggest changes was two new keywords, let and const, to create, or *declare*, [variables](https://www.codecademy.com/resources/docs/javascript/variables). Prior to the ES6, programmers could only use the var keyword to declare variables.

var myName = 'Arya';  
console.log(myName);  
// Output: Arya

Let’s consider the example above:

1. var, short for variable, is a JavaScript *keyword* that creates, or *declares*, a new variable.
2. myName is the variable’s name. Capitalizing in this way is a standard convention in JavaScript called *camel casing*. In camel casing you group words into one, the first word is lowercase, then every word that follows will have its first letter uppercased. (e.g. camelCaseEverything).
3. = is the *assignment operator*. It assigns the value ('Arya') to the variable (myName).
4. 'Arya' is the *value* assigned (=) to the variable myName. You can also say that the myName variable is *initialized* with a value of 'Arya'.
5. After the variable is declared, the string value 'Arya' is printed to the console by referencing the variable name: console.log(myName).

There are a few general rules for naming variables:

* Variable names cannot start with numbers.
* Variable names are case sensitive, so myName and myname would be different variables. It is bad practice to create two variables that have the same name using different cases.
* Variable names cannot be the same as *keywords*. For a comprehensive list of keywords check out [MDN’s keyword documentation](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical_grammar#Keywords).

In the next exercises, we will learn why ES6’s let and const are the preferred variable keywords by many programmers. Because there is still a ton of code written prior to ES6, it’s helpful to be familiar with the pre-ES6 var keyword.

If you want to learn more about var and the quirks associated with it, check out the [MDN var documentation](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/var).

### Create a Variable: let

As mentioned in the previous exercise, the let keyword was introduced in ES6. The let keyword signals that the variable can be reassigned a different value. Take a look at the example:

let meal = 'Enchiladas';  
console.log(meal); // Output: Enchiladas  
meal = 'Burrito';  
console.log(meal); // Output: Burrito

Another concept that we should be aware of when using let (and even var) is that we can declare a variable without assigning the variable a value. In such a case, the variable will be automatically initialized with a value of undefined:

let price;  
console.log(price); // Output: undefined  
price = 350;  
console.log(price); // Output: 350

Notice in the example above:

* If we don’t assign a value to a variable declared using the let keyword, it automatically has a value of undefined.
* We can reassign the value of the variable.

### Create a Variable: const

The const keyword was also introduced in ES6, and is short for the word constant. Just like with var and let you can store any value in a const variable. The way you declare a const variable and assign a value to it follows the same structure as let and var. Take a look at the following example:

const myName = 'Gilberto';  
console.log(myName); // Output: Gilberto

However, a const variable cannot be reassigned because it is constant. If you try to reassign a const variable, you’ll get a TypeError.

Constant [variables](https://www.codecademy.com/resources/docs/javascript/variables) must be assigned a value when declared. If you try to declare a const variable without a value, you’ll get a SyntaxError.

If you’re trying to decide between which keyword to use, let or const, think about whether you’ll need to reassign the variable later on. If you do need to reassign the variable use let, otherwise, use const.

### Mathematical Assignment Operators

Let’s consider how we can use [variables](https://www.codecademy.com/resources/docs/javascript/variables) and math [operators](https://www.codecademy.com/resources/docs/javascript/operators) to calculate new values and assign them to a variable. Check out the example below:

let w = 4;  
w = w + 1;  
  
console.log(w); // Output: 5

In the example above, we created the variable w with the number 4 assigned to it. The following line, w = w + 1, increases the value of w from 4 to 5.

Another way we could have reassigned w after performing some mathematical operation on it is to use built-in mathematical assignment operators. We could re-write the code above to be:

let w = 4;  
w += 1;  
  
console.log(w); // Output: 5

In the second example, we used the += assignment operator to reassign w. We’re performing the mathematical operation of the first operator + using the number to the right, then reassigning w to the computed value.

We also have access to other mathematical assignment operators: -=, \*=, and /= which work in a similar fashion.

let x = 20;  
x -= 5; // Can be written as x = x - 5  
console.log(x); // Output: 15  
  
let y = 50;  
y \*= 2; // Can be written as y = y \* 2  
console.log(y); // Output: 100  
  
let z = 8;  
z /= 2; // Can be written as z = z / 2  
console.log(z); // Output: 4

Let’s practice using these mathematical assignment operators!