**Interactive Development w/ JavaScript**

**Variables**

Variables are containers we will use to store information or data. Each variable we use will have an identifier (unique name). Some variables will be predefined in the language with an identifier already assigned, and other ones we will give names that are generally short and descriptive, often using more than one word without spaces between. An example might be a variable we give the name of finalGrade. When giving a variable name, there are general rules we need to follow, such as:

* Names must begin with a letter, with two exceptions, it may begin with a $ or an underscore. Variable names may not start with a numeric literal.
* Names may contain characters, such as letters, numeric literals, dollar signs, and underscores.
* Names are case sensitive such that the characters ‘A’ and ‘a’ would be referring to different variables.
* JavaScript will not allow reserve words to be used as variable names.

Variable declaration examples:

* var finalGrade = "A"; var gradeAverage = 93.2;
* var – declaration

finalGrade and gradeAverage – variable names/identifiers

* = – assignment operator, a topic discussed more in the next module
* "A" – string literal
* 93.2 – numeric literal ;
* - indicates the end of a JavaScript statement

**Variables Scope**

The scope of a variable determines the accessibility for its use; in other words, the area of code from where the variable can be used. This is the portion of the program that can refer to the declared variable by name. In JavaScript there are two types of scope, local and global.

* Local variables may be accessed from the point of declaration until the end of that block.
* Global variables are declared outside of blocks, such as functions. Functions will be discussed later in the course.

A block of code is a group of one or more statements enclosed in opening and closing curly braces, using a { to open a block and a } to close a block.

Local Variable Specifics:

* Inside all blocks, variables may be declared.
* Variables declared in different scopes may have identical names.
* Stored in memory during the running of a block.
* Generally, variables should be local when possible.

Global Variable Specifics:

* These variables are stored in memory for the entire time the application is running.
* All variables assigned a value without being declared (var) are global, regardless of the scope in which the value is given.

**Variables - Data Types**

JavaScript supports five different primitive variable data types:

* Numeric
  + 840
  + 62.31
  + -622
  + -45.8208
* Strings
  + Valid
    - "Hello World"
    - 'Hello World'
    - "Example 'Hello World' "
    - 'Example "Hello World" '
  + Invalid
    - "Hello World'
    - 'Hello World"
    - "Example 'Hello World" '
    - 'Example "Hello World' "
* Boolean
  + true
  + false
* Undefined
  + Variables without a value that exists.
  + Variables without a value that is declared.
* Null
  + Variable with an empty value, basically a placeholder holding nothing.

To summarize the difference between undefined and null, undefined has no value assigned, and null has an empty value assigned.

**Update Text and Markup**

There are several techniques for adding text and markup. Some are:

* Alert Box
  + The windows object’s alert ( ) method creates a dialog on top of a page.
  + Example:
  + window.alert("Window Alert");

​

* document.write
  + The document object’s write ( ) method adds content, not in the original document. It is advised the write ( ) method be avoided as:
    - Only works appropriately when a page is initially loaded.
    - If used after initially loading, it will overwrite the entire page.
    - Is not used often.
  + Example:

document.write("Welcome to JavaScript");

* element.innerHTML
  + The innerHTML property will allow you to update content of an element.
    - Will add new markings using less code than DOM manipulation.
    - Can be faster than DOM manipulation.
    - Simple way to remove element’s content.
    - Should not be used to add content from a user, such as a user’s name.
  + Example:

document.getElementById("par\_02").innerHTML "Welcome to JavaScript";​

* console.log
  + The console's log ( ) method write to the console and is generally used for debugging.
  + Example:

console.log("Console Log Text");​

DOM manipulation (topic to be covered later in this course)