| **Variable** | **Explanation** | **Example** |
| --- | --- | --- |
| **[String](https://developer.mozilla.org/en-US/docs/Glossary/String)** | This is a sequence of text known as a string. To signify that the value is a string, enclose it in single or double quote marks. | let myVariable = 'Bob'; or let myVariable = "Bob"; |
| **[Number](https://developer.mozilla.org/en-US/docs/Glossary/Number)** | This is a number. Numbers don't have quotes around them. | let myVariable = 10; |
| **[Boolean](https://developer.mozilla.org/en-US/docs/Glossary/Boolean)** | This is a True/False value. The words true and false are special keywords that don't need quote marks. | let myVariable = true; |
| **[Array](https://developer.mozilla.org/en-US/docs/Glossary/Array)** | This is a structure that allows you to store multiple values in a single reference. | let myVariable = [1,'Bob','Steve',10]; Refer to each member of the array like this: myVariable[0], myVariable[1], etc. |
| **[Object](https://developer.mozilla.org/en-US/docs/Glossary/Object)** | This can be anything. Everything in JavaScript is an object and can be stored in a variable. Keep this in mind as you learn. | let myVariable = document.querySelector('h1'); All of the above examples too. |

| **Operator** | **Explanation** | **Symbol(s)** | **Example** |
| --- | --- | --- | --- |
| **Addition** | Add two numbers together or combine two strings. | + | 6 + 9; 'Hello ' + 'world!'; |
| **Subtraction, Multiplication, Division** | These do what you'd expect them to do in basic math. | -, \*, / | 9 - 3; 8 \* 2; // multiply in JS is an asterisk 9 / 3; |
| **Assignment** | As you've seen already: this assigns a value to a variable. | = | let myVariable = 'Bob'; |
| **Strict equality** | This performs a test to see if two values are equal and of the same data type. It returns a true/false (Boolean) result. | [===](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Strict_equality) | let myVariable = 3; myVariable === 4; |
| **Not, Does-not-equal** | This returns the logically opposite value of what it precedes. It turns a true into a false, etc.. When it is used alongside the Equality operator, the negation operator tests whether two values are not equal. | !, !== | For "Not", the basic expression is true, but the comparison returns false because we negate it:  let myVariable = 3; !(myVariable === 3);  "Does-not-equal" gives basically the same result with different syntax. Here we are testing "is myVariable NOT equal to 3". This returns false because myVariable IS equal to 3:  let myVariable = 3; myVariable !== 3; |