# JS Built In Functions

Lesson Time: 30 Minutes

Now that you understand you can create your own functions, we need to look at built-in functions. A built-in function is a function that is “built into” every JS primitive data type. You don’t have to write them, the creator’s of JS have made them for you, and they live inside of every string, number, & boolean variable we’ve worked with so far. We also call them **methods. Any function that exists inside of a JS primitive or object we call a method.**

> This terminology can be very confusing. Functions and methods are pretty much the exact same thing. The only difference is that a function exists alone and is not attached to any data object. A method is just a function that lives inside a data object.

Built-In functions, or methods, give us a large set of tools for working for our variables. We use a methods by typing:

```
variableName.method()
```

**Common String Methods**

| Built-In Function | Purpose |
|---|---|
| split(characters to split on) | Split a string into two or more pieces on a certain character or letter. |
| replace(letters to replace, new letters) | Replace this letter or letters with another something else |
| slice(beginning, end) | Create a new string that starts here and ends there |
| indexOf(letters to search for) | Find the position of a letters in a string |
| ToLowerCase() | Convert the string to all lower case |
| ToUpperCase() | Convert the string to all uppercase |
| Includes() | Checks if the text contains a search string and returns true/false |

**Common Array Methods**

| Built-In Function | Purpose |
|---|---|
| reverse | Reverse the order of the array |
| pop() | Take the last value out of the array |
| push() | Add a new value to the end of the array |
| shift() | Take the first value out of the array.<br>Everything shifts down one index position. |
| unshift() | Add a new value to the beginning of the array.<br>Everything shifts up one index position. |

To best understand them, it's best to see them in action by downloading our example **built-in Functions.JS** file. Download the examples to see each one in use.

There are many more built-in functions than the ones listed here. A complete list is available at W3 Schools in the Resources section of the lesson.

Finally, in addition to methods in every string, number & array data type, there are also Global Functions and Math Methods that are also often used to process data.

| Key Terms | Methods, object methods, built-in functions |
|---|---|
| Lesson Files | Built-inFunctions.js |
| Additional Resources | Complete Reference<br>https://www.w3schools.com/jsref/jsref_obj_array.asp<br>https://www.w3schools.com/jsref/jsref_obj_string.asp<br>https://www.w3schools.com/jsref/jsref_obj_number.asp |
| Further Learning | |

# JS Variable Scope

Lesson Time: 30 Minutes

Variables have scope. **Scope** defines which sections of the script the variable may be used it.

Our first scope is **global** scope.  When a variable has global scope, it can be reused *everywhere*. To give a variable global scope, create it outside of any functions or code blocks.

The next scope is **function** scope. A function is a few small lines of reusable code, and is created with the function keyword. When a variable has function scope, it can *only be used inside of the function*.  The variable does not exist and can not be used outside of the function it is created in. To give a variable function scope, create it inside a function.

Our next scope is **block scope**.  A code block is enclosed in { }.  var and let act differently here.  var ignores code blocks, let does not.  This means a var created in a code block can be re-used outside of it. Variables created with let can not.

This is best illustrated with examples.  Look at the **let** tutorial on W3 schools to see code examples of how the variables behave.

https://www.w3schools.com/js/js_let.asp

| | |
|---|---|
| Key Terms | Global scope, functional scope, block scope |
| Lesson Files | |
| Additional Resources | https://www.w3schools.com/js/js_let.asp |
| Further Learning | |