Functions are useful so you don’t have to keep retyping code; you can instead simply invoke the function whenever needed.

An example could look like this:

https://techacademystorage.blob.core.windows.net/javascript/functions2.png

Other JavaScript code elements could call this “add” function by specifying its name and passing it two numbers. That could look like this:

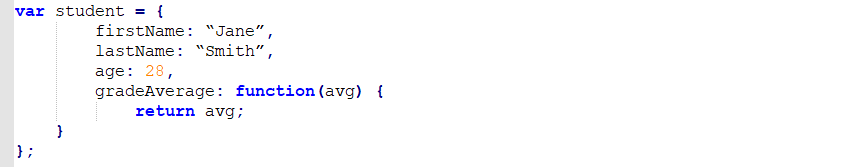
https://techacademystorage.blob.core.windows.net/javascript/functions3.png

Here, the code would create the variable called “sum”, call the function “add” and pass it the two numbers 5 and 7, and take the result (12) and assign that value to the variable “sum”.

**METHODS**

In JavaScript, a method is a set of code associated with an object that is designed to change the state of that object when it executes. In other words, the method is performed on the object.

You create these methods when you create the object. Let’s look at how we might do that with our previous example of a “student” object:



Here, we aren’t setting the property “gradeAverage” to a fixed number of 3.5. Instead, we are setting that property to the value returned by a set of code. Specifically, that code will take in a number (the variable “avg”) and set the value of the “gradeAverage” property to the value of that variable.

Executing that code could look like this:

https://techacademystorage.blob.core.windows.net/javascript/functions5.png

Here, we are telling the computer to make use of the object called “student”. Specifically, the computer is to run the function “gradeAverage” that is defined in the “student” object. Since that function needs an input (the variable “avg”), we give it the number 3.4.

So here’s the distinction: In this specific situation, that function “gradeAverage” is called a method. Yes, it’s confusing – until you recall that in JavaScript, a method is a set of code, associated with an object, that is performed on the object itself. So we have the confusing situation of a property of an object that we are defining using a function – but we’re calling the action performed by the function a method.

**LINKING JAVASCRIPT FILES**

Just like with CSS, our JavaScript code can be written in external files that can be linked to from our HTML file. To do so, we include the src attribute within a script tag, as follows:

https://techacademystorage.blob.core.windows.net/javascript/linking_javascript.png

So, to run an external file, we would write this code within the external file:

https://techacademystorage.blob.core.windows.net/javascript/linking_javascript2.PNG

Save the file as “JavaScript\_File.js.” Then, within the body or head text of our HTML document, we’d write:

https://techacademystorage.blob.core.windows.net/javascript/linking_javascript3.PNG

It is best practice to keep all JavaScript within an external file.

**ASSIGNMENT**

1. Create a new folder inside your Basic JavaScript Projects folder and name it “JS”.

2. Create a new file and write the alert() method.

3. Save this file as “Basic\_JavaScript\_1.js” within the JS folder.

4. Open your Basic\_HTML\_1.html and delete the script element.

5. Within the body or head write a script element containing a src attribute.

https://techacademystorage.blob.core.windows.net/javascript/linking_javascript4.PNG

6. Save the HTML file and then successfully execute it in the browser.

**WINDOW.ALERT() METHOD**

We can cause an alert window to pop up using the window.alert() method as well:

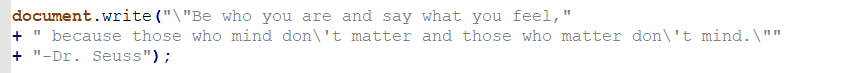
https://techacademystorage.blob.core.windows.net/javascript/methods1.png

**CONCATENATING A STRING**

“Concatenate” means to connect things together, like links in a chain. It means to take one piece of data and stick it on the end of another piece of data.

For example: concatenating the string “device” and the string “s” makes the text “devices.”

To concatenate a string in JavaScript, you use the + operator as follows:



This text would display as, "Be who you are and say what you feel, because those who mind don't matter and those who matter don't mind."-Dr. Seuss

**CHALLENGE**

Assign a concatenated string value to a variable and display it in the browser.

HINT:

https://techacademystorage.blob.core.windows.net/javascript/strings3.PNG

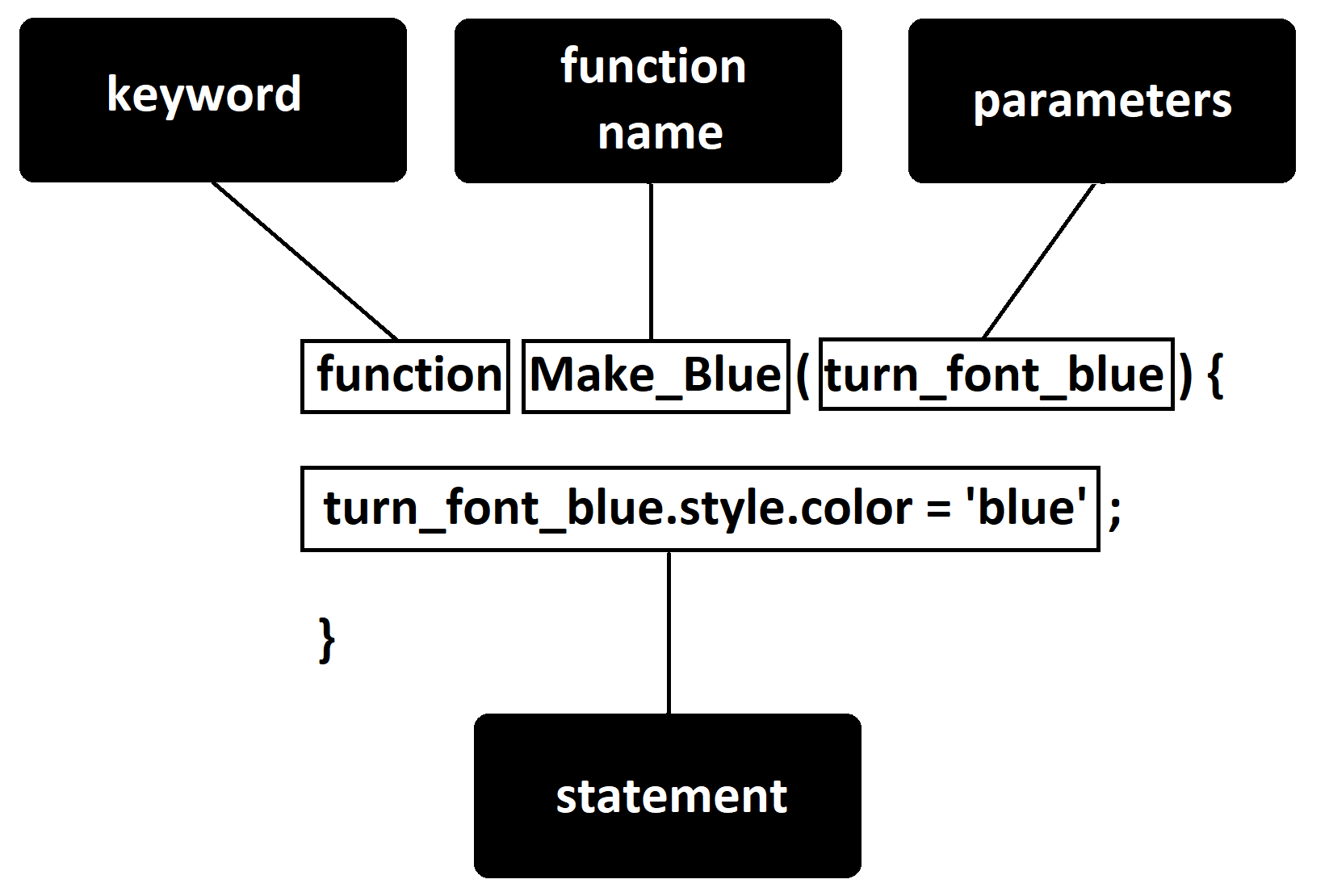
**FUNCTIONS**

As a recap, a JavaScript function is a repeatable block of code that executes certain actions and performs tasks. You execute a function by calling it. This is also called “invoking” the function.

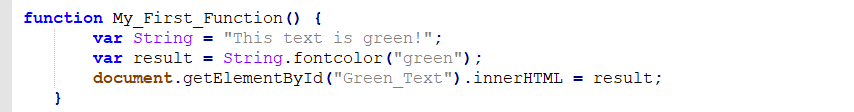
Functions are valuable because of code reusability – you can invoke functions over and over. Since functions in JavaScript contain properties and methods, they are basically objects.

“document.getElementById” is a method that returns an element. The element has an ID attribute with a specific value assigned to it. It is used mainly to control or get information from an element within your code. If it can’t find the element with the specified value, it will return “null.”

In JavaScript, a keyword identifies actions to be performed. There is a function keyword. Parameters are the values passed to or received by the function. One writes a JavaScript function with the keyword, then a name, then parentheses containing parameters (note: you can leave the parentheses empty).

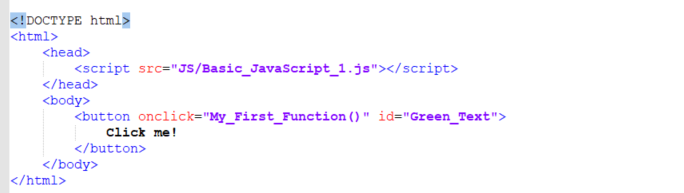


Here is an actual function (this is JavaScript code – it would be written within the external .js file):



Everything inside our curly brackets { } is our function.

The HTML code within our .html file would be written as follows:



In this HTML code, we created a button element. We then utilized the ID attribute and assigned the button element the value “Green\_Text.”

The button element was also given an event handler which is used to call the JavaScript function “My\_First\_Function()”. The onclick event handler is triggered when the user clicks on that element.

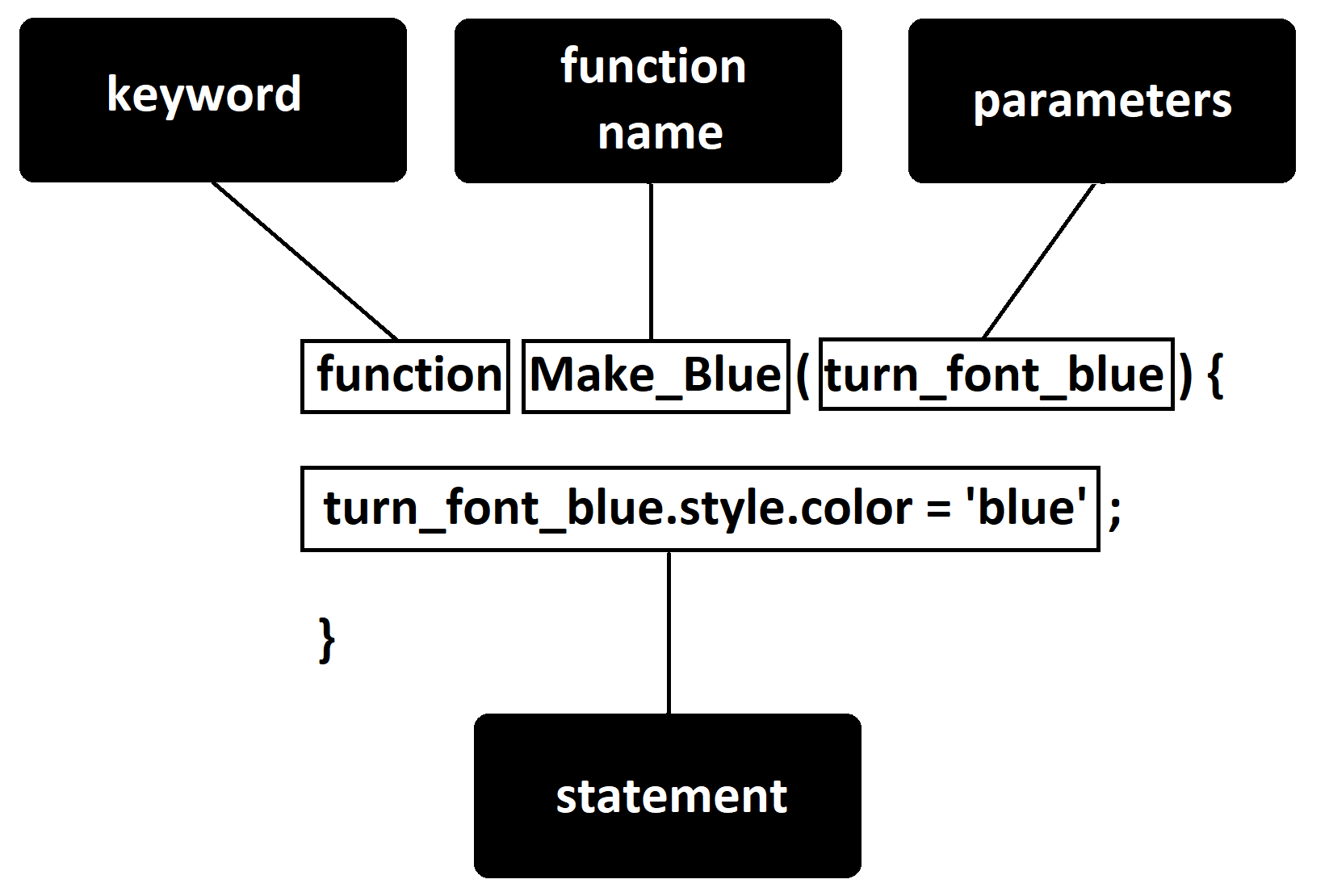
**FUNCTIONS**

As a recap, a JavaScript function is a repeatable block of code that executes certain actions and performs tasks. You execute a function by calling it. This is also called “invoking” the function.

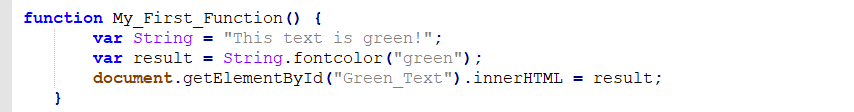
Functions are valuable because of code reusability – you can invoke functions over and over. Since functions in JavaScript contain properties and methods, they are basically objects.

“document.getElementById” is a method that returns an element. The element has an ID attribute with a specific value assigned to it. It is used mainly to control or get information from an element within your code. If it can’t find the element with the specified value, it will return “null.”

In JavaScript, a keyword identifies actions to be performed. There is a function keyword. Parameters are the values passed to or received by the function. One writes a JavaScript function with the keyword, then a name, then parentheses containing parameters (note: you can leave the parentheses empty).

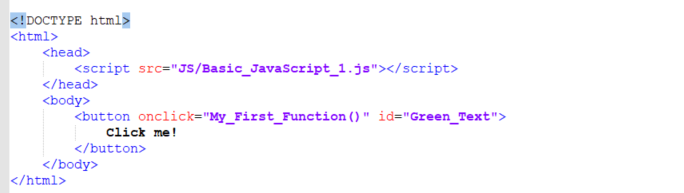


Here is an actual function (this is JavaScript code – it would be written within the external .js file):



Everything inside our curly brackets { } is our function.

The HTML code within our .html file would be written as follows:



In this HTML code, we created a button element. We then utilized the ID attribute and assigned the button element the value “Green\_Text.”

The button element was also given an event handler which is used to call the JavaScript function “My\_First\_Function()”. The onclick event handler is triggered when the user clicks on that element.

Suppose you wanted to find all paragraph elements in an HTML document. You might use code like this:



The variable called “foo” would now contain a collection of all the paragraph tags in the document.

