# NM2207

Week 4

Session 04 Classwork

Overview of what we’ll do today:

* Practice using the console.log function call
* Practice with different kinds of functions
* Practice using objects

Before attempting the challenges, you are expected to have watched and coded along with the Lecture videos. A tutorial is meant to practice the skills presented in the video lecture and show you more applications of it. Tutors will explain the challenges and answer the questions you may have**.**

**The challenges are due to be completed at the end of class each week for full credit which is also attendance. Submitting the next day accounts for half the credit.**

As usual, open the index.html and main.js for Session04.class in Visual Studio Editor (VSCode). At the end of the tutorial, use the last half an hour to ensure everything is uploaded to Filezilla and linked to your homepage. This is essential to get your grade for classwork.

You may see concepts below that build on what we did in the videos or read about in the textbook. This is normal and expected. Do not be alarmed.

Unlike other modules, in a tutorial, programming exercises are never “exactly” what we did in the lecture, and always push you to try new things.

In the Quiz, there will be no such traps. Only something that’s in a video lecture, or covered in the tutorial, or covered in Grasshopper, will be assessed.

Summary of learnings at the end of this part:

* If you create a new function, you also have to make sure that you call it otherwise it will not execute.
* If it takes in arguments, then you have to pass it arguments when you call it.
* If it returns a value, then you have to assign it to a variable when you call it.
  + Eg var myFactorialValue = myFactorial(5);

# Reflection / discussion (10 minutes)

1. What is the basic anatomy of a function?
2. How is a function different from the statements we’ve been writing so far?
3. How can you use console.log to debug a function?

There will be at least one such question in the Quiz. You don’t need to include these answers (for a – c) in your classwork today, but take the time to make notes and be clear on how you would answer them. Use the textbook and other resources to make sure you are accurate.

# Warmup (10 minutes)

The console.log() is actually a function call, where we call “the log function” of “the console window” and pass it some values ie whatever is within the round brackets. It is awesome for troubleshooting, but we need to understand its syntax ie the rules of calling it correctly.

Paste each of the statements in Table 1 in your main.js after the two variables (prodValue and alsoValue) are declared. Save, and reload the index.html file in your Chrome browser.

**You might need to fix the double quotes when you paste in the statements by deleting the double quote symbols and manually creating them using shift key + quotes key. Ask your partner or the TA if you’re not sure how to get the double quotes.**

Open the console window and record the outputs on the console log. Complete the Table. This is for self-reference. We do not need you to upload it to your classwork today.

|  |  |  |
| --- | --- | --- |
| Statement | Output | Explain why |
| console.log("Row 1: prodValue + alsoValue"); |  | Inside quotes so will literally display the stuff |
| console.log(“Row 2: ” +prodValue + alsoValue); |  | Concatenates strings and numbers |
| console.log(prodValue + alsoValue); |  | Adds numbers |
| console.log("prodValue + alsoValue" + prodValue + alsoValue); |  |  |
| console.log("prodValue + alsoValue" + " " + prodValue + " " + alsoValue); |  |  |
| console.log(“Will this work: “ + alsovalue); |  |  |
| console.log(“Will this work: “ alsovalue); |  |  |

**Part 1**

**Writing functions in JavaScript (40 minutes)**

A function encapsulates a set of commands that we want should be executed together. It can optionally take in some input in the first line, and it can return some output in the last line.

Putting statements and expressions into functions can make code more modular, less prone to errors, and easier to debug.

**Functions with no arguments and no return value**

* 1. Open Session04.class/appscripts/main.js
  2. Paste the code in firstbox.txt provided in the resources folder, into main.js
  3. Be careful to paste it before the function it defines is called in main.js.
  4. Save. Open Session04.class/index.html in the browser, and use the developer tools to open the console window.
  5. What happens when the page loads? Why?
  6. What happens if you pasted the code after the function call, save, and then reload? Why?
  7. What happens when you try to pass it arguments? Why?
  8. Write your answers for 1.5 – 1.7 in the index.html for Session04.class under a heading “Part 1: Functions with no arguments and no return values”. No screenshots are needed. At the end of the tutorial, you will be required to show this (uploaded thru filezilla and linked to your nm2207 website) as proof of classwork, to your tutors.

**Functions with two arguments and no return value**

* 1. Modify the function definition so that it takes in 2 arguments, a and b.
  2. Now, what happens when you call it without passing it arguments? Why?
  3. What happens when you call it with two arguments? Why?
  4. In the function definition, change “prod = p1\*p2;” to “prod = a\*b;”
  5. Now, what happens when you call it with two arguments? Why?
  6. Write your answers for 1.10 – 1.13 in the index.html for Session03.class under a heading “Part 1b: Functions with two arguments and no return value”. Also include screenshots of your main.js and the console.log outputs for each of them. At the end of the tutorial, you will be required to show this (uploaded thru filezilla and linked to your nm2207 website) as proof of classwork, to your tutors.

**Function with two arguments and one return value**

* 1. Modify the function definition by adding the line “return prod;” to the last line of the function, before the closing curly bracket.
  2. If a function returns a value, we want to store it when we call the function. So, modify your function calls by assigning them to a variable, like “var answer = myFunction(5,6);”
  3. Is this working as expected? Comment out the existing console.log statements, and create one to print the new variable you just defined (answer) to the console log.
  4. What is the “return” statement doing?
  5. Write your answers for 1.16 – 1.18 in the index.html for Session03.class under a heading “Part 1c: Functions with two arguments and one return value”. No screenshots are needed. At the end of the tutorial, you will be required to show this (uploaded thru filezilla and linked to your nm2207 website) as proof of classwork, to your tutors.

**References:**

1. <https://www.w3schools.com/js/js_functions.asp>
2. <https://www.geeksforgeeks.org/javascript-console-log-with-examples/>

**(Suggested 10 minute break)**

**Part 2 Objects (50 minutes)**

**Returning an object instead of a single value (15 minutes)**

* 1. Create a new function "calculator" that takes 2 numbers as arguments, and calculates their sum, difference and product, and stores them in three variables.
  2. Now, within the function definition, define a variable with the name result. It should have three properties, “sum”, “diff” and “prod”.
  3. Assign your calculated outputs from 3.2 as the values of the three properties of your new object, result.
  4. Return result at the end of the function definition, before the closing curly bracket
  5. Now, calculator is a function which accepts two arguments and return an object which has 3 properties, each of which reflects one of the 3 operations that it calculates.
* Call the calculator function. Since it takes in some parameters and returns a value, how would you need to call it?
  1. Take the object returned by the function call. Print its properties (something.sum, something.diff, something.prod) to the console.
  2. Print its properties to the webpage by modifying the innerHTML of a webpage element. The message should start as “**Output of Step 2.7 is:”**
* Take a moment to add comments to your code.

**(Suggested 10 minute break)**

**Part 3 (30 minutes)**

**Variable scope**

**Summary of learnings**

Variable scope (the difference between variables declared inside function and those declared outside functions)

* ***Variable Declaration:*** *Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the var keyword or the let keyword.*
* ***Variable Initialization:*** *This usually occurs when a variable is declared. Here the variable is assigned a memory or space by the JavaScript engine. Because of this, once a variable is declared, it takes a value of undefined even before assignment.*
* ***Variable Assignment:*** *Variable assignment is usually the most important step when using a variable. Here the variable is assigned data which is a value using the assignment operator "=". Values in JavaScript take one of the standard JavaScript datatypes (string, number, boolean, null, undefined)*

**Reference:** https://www.tutorialspoint.com/javascript/javascript\_variables.htm

**Long-term and short-term memory loss with variable scope (20 minutes)**

3.1 Let’s play with a new code snippet. It is provided as part2.txt in the resources folder. Notice the pre-function block of statements before the function, the function block statements within the function, and the post-function block.

//PRE-FUNCTION BLOCK

var counter2; //LINE 1: declaring a variable. Current value = undefined  
counter2 = 5; //LINE 2: assigning a variable. Current value = 5

function foo() {

//FUNCTION BLOCK

counter2 = counter2 + 1;  
 console.log(“Inside the function block: counter2’s value is:” + counter2);

}

//POST-FUNCTION BLOCK

console.log(“Post-function block: counter2’s value is:” + counter2);

foo();

3.2 Now, to understand variable scope, we will edit this code. Shuffle Line 1 and Line 2 between different positions in the pre-, post- and the function block. Remember that Line 1 should always occur before Line 2!

3.3 In other words, move your declaration and assignment of counter2 so that you try out all the conditions in the Table 2.

**Please fill up the following table with your findings. Note the ORDER in which console.log statements are printed. Why is this so?**

**Table 2**

|  |  |  |
| --- | --- | --- |
|  | **Output on the console** | **Why** |
| **Line 1 and Line 2 are in current position** |  |  |
| **Line 2 is inside function block BEFORE “counter2 = counter2+1” expression** |  |  |
| **Line 2 is inside function block AFTER “counter2 = counter2+1” expression** |  |  |
| **Line 2 is in post-function block, after function call** |  |  |

Write your answers for 3.3 in the index.html for Session03.class under a heading “Part 3: Variable scope”. No screenshots are needed. At the end of the tutorial, you will be required to show this (uploaded thru filezilla and linked to your nm2207 website) as proof of classwork, to your tutors.

**Part 4 Initializing objects (30 minutes)**

* 1. Create two variables with names 'point1' and point2'
* Imagine that point1 and point2 are two points on the x,y plane. That is, both of them have an x property, and a y property.
* Define point1 and point2 with any numeric properties for point1.x, point1.y, point2.x, and point2.y Explore the Eloquent JavaScript textbook (Pg 61 and onward) to see how this is done.
* Print point1 and point2 to the console. Do you see what you expected to see?

**References:**

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects>

**Operations on object properties (15 minutes)**

4.2 Create a new function called 'pointsum', that expects two points as input arguments (that is, two objects each with an 'x' and a 'y' property). The function should return a new point object that is the sum of the two input points.

* The sum is calculated by summing the individual x and y properties of the two arguments to each other.

4.3 As before, add sentence to your article that expresses the results of calling your function. The message should start as “**Output of Step 4.3 is**:”

* Hint: JSON is a built-in object that has a stringify method that "pretty prints" objects provided as an argument. Use JSON.stringify() to print the result of the call to your pointsum function.
* Take a moment to add comments to your code.

**References**

<https://levelup.gitconnected.com/5-ways-to-log-an-object-to-the-console-in-javascript-7b995c56af5a>

**Reflection and discussion**

* What would an object look like that has a function as a property?
* How would you call a function that is a property of an object?

**Part 5 Bonus**

**More practice**

* 1. **Passing arguments to functions (10 minutes)**

5.1.1 In your main.js, add a new function named 'sumMaker ' that takes no arguments. All this function should do is print a console message saying that you are in the function.

* First, create an empty function, similar to the one in the first box.
* Then, fix its name from myFunction to sumMaker,
* Then, remove anything in the round brackets because it takes no parameters.
  + 1. Reload the page. Check the console window. The function will not be executed, because you have to call a function for it to be executed. So far you have only defined it.
* Now, add a function call to main.js after the statements that define the sumMaker function, similar to the last few lines in the first box. Save and reload in the browser. Does it work now?
  + 1. Now modify sumMaker so that it takes two numbers as arguments, prints a console message saying you are in the function.

**You can print the outputs to the console log by including this statement in the sumMaker function:**

**console.log(“Output of Step 5.1.3 is : I am in the function“);**

* + 1. Save and reload. It will throw an error because it expected arguments, but your function call is not passing any arguments to it.

Before you call sumMaker, create two variables, 'x', and 'y', and assign numbers to them. Now use them in the sumMaker() call e.g., sumMaker(x,y);

Does it work now? Take a moment to add comments to your code.

* 1. **Saving the output from a function into a variable (15 minutes)**

5.2.1 Now create a function sumMaker2 that does the exact same thing as before, but now it also returns the difference between the two numbers it received as arguments.

Now you will have to change the way you call the function. When you call the function, store the difference in another variable.

* + 1. Use the variables and the value returned by calling the 'sumMaker2' function on them to construct a sentence printed in your article element. The sentence should read something like: **"Output of Step 5.2.2 is: The difference between 7 and 100 is 93"**
  + Hint: if you don't want to overwrite the text already in your article element, put a line break ( <br>) at the end of your previous welcome message, then use '+=' to add the new text.
  + Take a moment to add comments to your code.