



LO7 READING NOTES

YAMILETH RIVERO GARCÍA

Web Frontend Development II

1.- CH11: FURTHER FUNCTIONS

In JavaScript, functions are first-class objects.


- They can be passed around in the same way as every other value.

Function Properties and Methods

Functions can have properties and methods themselves.


Example: All functions have a **length** property that returns the number of parameters the function has.

If we query the **length** property, we can see that it accepts one parameter



```
function square(x) {  
  return x*x;  
}
```

```
square.length  
<< 1
```




Call and Apply Methods

The `call()` method can be used to set the value of `this` inside a function to an object that is provided as the first argument.

Example:


The `sayHello()` function refers to an unspecified object called `this` that has a property called `name`:

```
function sayHello(){  
    return `Hello, my name is ${ this.name }`;  
}
```



```
const clark = { name: 'Clark' };  
const bruce = { name: 'Bruce' };  
  
sayHello.call(clark);  
<< 'Hello, my name is Clarke'  
  
sayHello.call(bruce);  
<< 'Hello, my name is Bruce'
```

We can create some objects that have a name property, then use the `call()` method to invoke the `sayHello()` function, providing each object as an argument. This will then take the value of `this` in the function:




Custom Properties

There is nothing to stop you adding your own properties to functions in the same way that you can add properties to any object in JavaScript.

Example: You could add a description property to a function that describes what it does:

```
square.description = 'Squares a number that is provided as an argument'  
<< 'Squares a number that is provided as an argument'
```






Immediately Invoked Function Expressions

An **Immediately Invoked Function Expression** – or **IIFE** – is an anonymous function that, as the name suggests, is invoked as soon as it's defined.

- This is easily achieved by placing parentheses at the end of the function definition

Note: The function also must be made into an expression, which is done by placing the whole declaration inside parentheses:



```
(function(){  
  const temp = 'World';  
  console.log(`Hello ${temp}`);  
})();  
<< 'Hello World'
```

Recursive Functions

A **recursive function** is one that invokes itself until a certain condition is met. It's a useful tool to use when iterative processes are involved.

```
function factorial(n) {  
  if (n === 0) {  
    return 1;  
  } else {  
    return n * factorial(n - 1);  
  }  
}
```

Example: A function that calculates the factorial of a number

Closures

Closures are one of JavaScript's most powerful features, but they can be difficult to get your head around initially.



2.- CH8: TRANSFORM AND TRANSITIONS

For years, the only way to display text on an angle was to use an image of text created in an image-editing program and the only way to animate was to change positioning with JavaScript!



TRANSFORMS

- ♥ The CSS3 transform property lets you translate, rotate, scale, and/or skew any element on the page
- ♥ We can manipulate an element's appearance using **transform functions**

Translation

- ♥ Translation functions allow you to move elements left, right, up, or down



The **translate(x,y)** function moves an element **x** from the left, and **y** from the top:

```
transform: translate(45px, -45px);
```

TRANSITION

Steps to create a simple transition using only CSS

- ♥ Declare the original state of the element in the default style declaration.
- ♥ Declare the final state of your transitioned element; for example, a **:hover** state





- ♥ Include the transition functions in your default style declaration using the transition properties, including: **transition-property**, **transition-duration**, **transition-timing-function**, and **transition-delay**. We'll look at each of these and how they work shortly.

The important point to note is that the transition is declared in the default or originating state.

