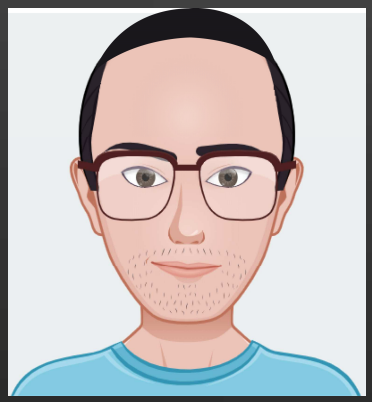
FORMATION FULL STACK

JAVASCRIPT – CHAPTER 3

Taking it Further

By: Meir Sabbah



Meir Sabbah

meir@promermedia.com

058-688-0761

Node Creation

```
<div class="allCars" id="mainCars"></div>
```

```
addCar(i);

function addCar(i) {
    var mainDiv = document.getElementById("mainCars");
    var newDiv = createElement("div");
    newDiv.id = "car" + i;
    newDiv.className = "eachCar";
    mainDiv.appendChild(newDiv);
}
```

What is THIS?!?

this

Is always an object

1) JS object's representation within.

this: 1) Object's representation within.

```
var user = {
    fullName: { lastName: "Bob", lastName: "Roberts" },
    ID: 123456789,
    DOB: "21/12/1980",
    creditCards: [{ visa: 418...789, exp: "08/21", cvv: 456 },
    { master: 437...384, exp: "12/19", cvv: 665 },
    { amex: 538...301, exp: "04/24", cvv: 7908 },
    qtyCreditCards: function () {
        return user.creditCards.length;
    }
};
```

this: 1) Object's representation within.

```
var user = {
    fullName: { lastName: "Bob", lastName: "Roberts" },
    ID: 123456789,
    DOB: "21/12/1980",
    creditCards: [{ visa: 418...789, exp: "08/21", cvv: 456 },
    { master: 437...384, exp: "12/19", cvv: 665 },
    { amex: 538...301, exp: "04/24", cvv: 7908 },
    qtyCreditCards: function () {
        return this.creditCards.length;
    }
};
```

What is THIS?!?

this

Is always an object

1) JS object's representation within.
2) HTML element representation.

this: 2) HTML element representation

```html
<div class="eachCar" id="car0">
    <h3 class="category">Economy</h3>
    <button class="btn" onclick="deleteCar(this)>X</button>
    <img src="assets/images/honda.jpg" alt="">
    <p class="make">Honda</p>
    <p class="model">Civic</p>
    <p class="price">$18,000</p>
</div>

<script src="js/main.js"> </script>
```

```javascript
function deleteCar(car) {
    var div = car. parentElement;
    div.innerHTML = "";
};
```

EXERCICES & TP

EXERCISE # 9

JS :

- Using your current "cars" project, create a function that will delete the item entirely, including from the array.

EXERCICES & TP

EXERCISE # 10

HTML :

- Create a form that is hidden, and can only be displayed when a button is clicked.
- Once open, that same button should say "Hide Form" and the form gets hidden when pressed.
- The form will contain a field for each "car" detail.

JS :

- What it takes to toggle the button and form functions.
- Create a function that will take in the input values and add that car to the existing list in the HTML.
- Make sure that the form gets cleared once submitted.

EXERCICES & TP

EXERCISE # 10

HTML :

- Créez un formulaire qui est masqué et ne peut être affiché que lorsque vous cliquez sur un bouton.
- Une fois ouvert, le même bouton devrait indiquer "Masquer le formulaire" et le formulaire est masqué lorsque vous appuyez dessus.
- Le formulaire contiendra un champ pour chaque détail de "car".

JS :

- Ce qu'il faut faire pour "toggle" les fonctions de bouton et de l'apparition du formulaire.
- Créez une fonction qui prendra les valeurs d'entrée et ajoutera cette voiture à la liste existante dans le code HTML.
- Assurez-vous que le formulaire est effacé une fois soumis.

Declared, Operated, & Anonymous **FUNCTIONS**

Function Declarator

```
sayHello();

function sayHello() {
    alert("Hello");
};
```

Function Operator

```
var sayHello = function() {
    alert("Hello");
};

sayHello();
```

Opposite won't work!

Anonymous Functions

```
var sayHello = function sayHi() {
    alert("Hello");
};

sayHello();
```

It will work, but `sayHi()` is **NOT** in scope, therefore is undefined!

Anonymous Functions

Self-Invoking Functions

```
(function() {
    alert("Hello");
})();
```

Created, called, and gone.

Anonymous Functions

Passed as an Argument

```
function add(calculate) {
    alert(calculate);
}

add(2 + 2);
```

Anonymous Functions

Passed as an Argument

```
function add(calculate) { //no args here
    alert(calculate(2, 2));
}

add(function(a, b) {
    return a + b;
});
```

Browser Object Model

Methods

`setTimeout()`
`setInterval()`

Delays or repeats a function call

Delayed Function

setTimeout()

```
setTimeout(function() {
    alert("Hello");
}, 2000);
```

Takes in two parameters, a function, and the quantity of milliseconds

Asynchronous Programming

```
console.log("Hello # 1");

setTimeout(function() {
    console.log("Hello # 2");
}, 2000);

console.log("Hello # 3");
```

Asynchronous Programming

```
for (let i = 1; i < 6; i++) {
    setTimeout(function() {
        console.log(i);
    }, 1000);
};
```

What happens here?

Repeated Function

setInterval()

```
setInterval(function() {
    alert("Hello");
}, 3000);
```

Takes in two parameters, a function, and the quantity of milliseconds

Goes on forever...

Stopping a Repeated Function

Introduction:

```
var thePrompt = prompt("Hello");
```

The prompt opens and receives a value, then the variable becomes that value.

Stopping a Repeated Function

Introduction:

```
var theAlert = alert("Hello");
```

The alert is run, but right after that, the variable becomes "undefined" because it finished it's job.

Stopping a Repeated Function

clearInterval()

```
var repeat = setInterval(sayHello, 3000);

function sayHello() {
    alert("Hello");
}


function stopIt () {
    clearInterval(repeat)
}
```

EXERCICES & TP

EXERCISE # 11

Make a stopwatch!