ES6

ECMAScript 6

Lecture 1

Agenda

- What is ESX ?
- Babel (Getting Started)
- Variable Declaration
- Arrow Functions
- Functions and Default Parameters
- Template Literals
- Object Literals Extensions
- The Rest Operator
- The Spread Operator
- The for-of Loop
- Destructuring
- Questions !

What is ECMAScript?

- ES12 is the latest version of ECMA.
- ECMA International.
- ES6, ES2015 and harmony are used to mean almost the same thing.
- Know the code and don’t concentrate on the versions.

Babel

- Not all features of ES6 are supported by all browsers at the moment.
  - https://kangax.github.io/compat-table/es6/
- We need transilers that takes the new easier ES6 that we are learning and convert it to ES5 (Old JS) that all browsers understand.
  - https://babeljs.io/repl/

Variable Declarations

- Declaring variables in ES5 using the var keyword (Global Scope).
- Both let and const have a block scope.
- The keyword key const is used for values that you don’t intend to change.
- In the old convention, we used to name the constant in all CAPS.

```
let x = 4;
if(x == 4) {
  let x = 3;
  console.log(x);
}
console.log(x);
```

Arrow Functions

- ES6 introduces a new kind of function called ‘Arrow Function’.
- It can replace any other ES5 function by doing some steps and changing the syntax a bit.

```
const getName = () => 'name';
```

Expressions vs Declaration

```
// Declaration function
function sayHello() {
    console.log('Hello');
}

// Expression function
const sayHello = function() {
    console.log('Hello');
}
```

Arrow Functions

- Arrow functions can only be used as an expression and can't be used as declaration.
    - You can store them in variables.
    - You can pass them as a callback function.

Arrow Functions (Steps)

1. Remove the function keyword.
2. Remove the return keyword.
3. Remove the semicolon.
4. Add an arrow (=>) between the parameter list and the function body.

Arrow Functions

- If the function takes one parameter we can remove ‘()’.

```
const myFunc = name => {
    console.log(name);
}
```

Arrow Functions

- If the body of the function is one statement only, we can remove the curly braces and in that case we don’t need a return keyword.

```
const myName = () => ‘Mohammed Mahmoud’;
// ES5
const es5Square = function(num) {
    return num * num;
}
// ES6
const es6Square = num => num * num;
```

Default Params

- ES6 make it possible to assign default parameters to functions to get a value instead of undefined.

```
const greet = (greeting = 'Hello') => {
    console.log(greeting);
}
greet() // Hello
```

Default Params

- Default parameters must come at the end of the params list.

```
const multiply = (a, b, c = 2) => a * b * c
console.log(multiply(5, 6)) // 60

const multiply2 = (x = 2, y, z) => x * y * z
console.log(multiply2(5, 6)) // NaN
```

Default Params

- Default parameters are available to later default parameters.

```
const support = (name = 'Mohammed', body = `I
support ${name}`) => {
    console.log(body);
}
```

Template Literals

- If we had an array of name as:

```
const NAMES = ["Ahmed", "Ali", "Hassan"]
```

- And we want the output to be like that:
  - Ahmed said: “I want to go on a trip with Ali and Hassan”

HOW TO DO THAT ?!

Template Literals

- They are strings that include in them some variables you want to embed.
- They written between back quotes ``;
- The expression you want to write will be written inside

```
${someVariable}
```

Object Literals

```
let name = "Ali";
let age = 25;
let ageField= "age";

let obj = {
    name,
    [ageField]: 26,
    “greet me“() {
        console.log(this.name + ' '+ this.age);
    }
};
console.log(obj[“greet me”]);
```

The Rest Operator

- The rest operator, written with three consecutive dots (...).
- Allow you to represent an indefinite number of elements as an array.

```
let numbers = [1, 2, 3, 4, 5]
function sumUp(toAdd) {
  let result = 0
  for (let i = 0; i < toAdd.length; i++) {
    result += toAdd[i]
  }
  return result
}
console.log(sumUp(numbers))
console.log(sumUp(100, 10, 20))
```

The Spread Operator

- The spread operator, also written in three consecutive dots(…).
- Allow you the ability to expand or spread, iterables into multiply elements.

```
const arr = [1, 2, 3, 4]
console.log(...arr) // 1 2 3 4
```

The For … Of

- In ES5 we used to loop over arrays with simple for loops or more complicated for … in loop.

```
let names = ["ahmed", "mohamed", "ibrahim"]

for (index in names) {
  console.log(names[index])
}
```

The For … Of

- In ES6 the For … Of loop was introduced to handle interables and its syntax was much simpler by dropping the index.
- It also can deal with strings (go throw each letter at a time).

```
let names = ["ahmed", "mohamed", "ibrahim"]

for (name of names) {
  console.log(name)
}
```

The Destructuring

- Destructuring is the act of extracting values from arrays and objects in just a single step (Very common used).

```
let numbers = [1, 2, 3]
let a = numbers[0]
let b = numbers[1]
let c = numbers[2]

let [a, ...b] = numbers
```

The Destructuring

- Destructuring can be done to objects also but with a very simple twist:

```
let person = {
  name: "Mohamed",
  age: 25,
}
const { name, age } = person
```

ANY QUESTIONS ?

LAB 1

- In the JS file, you will find a ‘web designer’ object.

  1- Change the ‘Your Name’ placeholder to your name.

  2- Write a getAge() function that takes the years alive array and return your age, save the value you return in const of name age.

  3- Divide the web designer skills into 2 variables ‘designSkills’ and ‘developmentSkills’ using (ES6).

LAB 1

4- Uncomment the newSkills array and merge the developmentSkills array with the newSkills array in a new array ‘updatedDevSkills’.

5-Destructure the diet array and using new ES6 write a function (getDrinks) that takes the newly created drinks variables and returns drinks that contain the letter ‘t’, call that function and save the returned values in a variable ‘tDrinks’.

LAB 1

6- Uncomment the function buildID().

In the return part of the function you will find a template literal.

Replace the placeholder data with you actual data:

Name => name

Age => age

Design Skills => designSkills

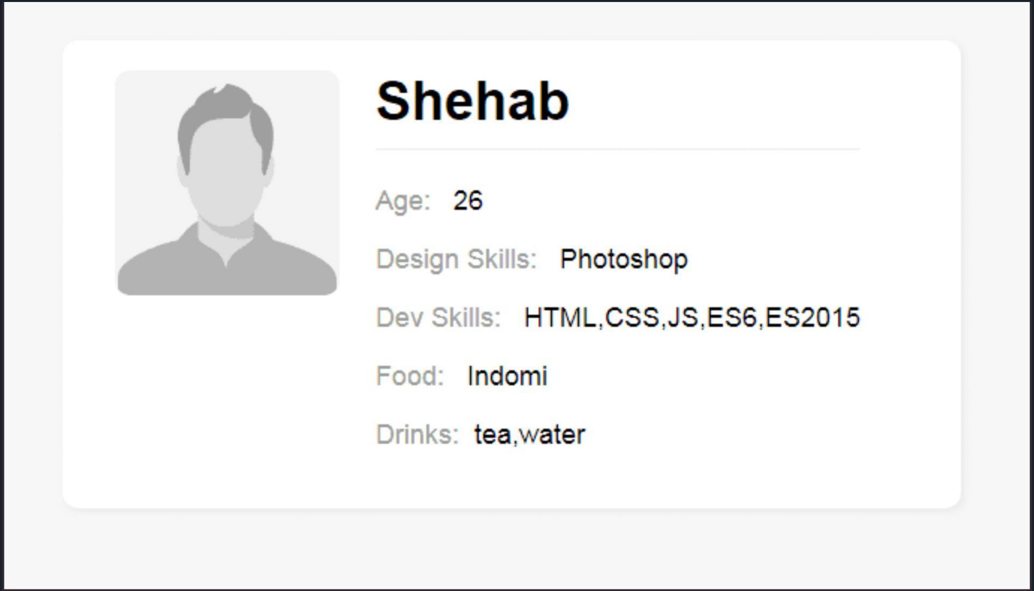
Dev Skills => updatedDevSkills

Food => Food

Drinks => tDrinks

LAB 1

7- Add the HTML Fragment you create to the div with class ‘card’.



Shehab

Age: 26

Design Skills: Photoshop

Dev Skills: HTML,CSS,JS,ES6,ES2015

Food: Indomi

Drinks: tea,water