Chapter 12 - Advanced JavaScript

There are some JavaScript concepts which make the life of a developer extremely simple. We will discuss some of those in this chapter

IIFE

IIFE is a JavaScript function that runs as soon as it is defined.

```
( function () {
    ...
    ...
  } )();
```

⇒ IIFE Syntax

It is used to avoid polluting the global namespace, execute an async-await, etc.

Destructuring

Destructuring assignment is used to unpack values from an Array, or properties from objects, into distinct variables

```
let [x, y] = [7, 20]
```

x will be assigned 7 and y, 20

```
[10, x, ...rest] = [10, 80, 7, 11, 21, 88]
```

x will be 80 rest will be [7, 11, 21, 88]

Similarly we can destructure objects on the left hand side of the assignment

```
Const obj = { a: 1, b: 2 }
const {a, b} = obj;
```

Some more examples can be found on MDN docs.

Spread Syntax

Spread syntax allows an iterable such as an array or string to be expanded in places where zero or more arguments are expected. In an object literal, the spread syntax enumerates the properties of an object and adds the key-value pairs to the object being created.

Example:

①
```
Const arr = [1, 7, 11]
Const obj = { ...arr }; // { 0: 1, 1: 7, 2: 11 }
```

②
```
Const nums = [1, 2, 7]
Console.log (Sum (...nums)) // 10
```

Other examples can be found on MDN docs

Quick Quiz: Output of the following ??

```
Const a = "the", b = "no"
Const c = { a, b }
Console.log (c)
```

Date
Page

local, global & block scopes

JavaScript has three types of scopes:

1. Block scope
2. Function Scope
3. Global Scope

let & const provide block level scope which means that the variables declared inside a { } cannot be accessed from outside the block

```
{
  let a = 27;
}
// a is not available here
```

Variables declared within a JavaScript function, become local to the function

A variable declared outside a function, becomes global

Hoisting

Hoisting refers to the process whereby the interpreter appears to move the declarations to the top of the code before execution

Variables can thus be referenced before they are declared in JavaScript

hello("Harry")

function hello(name) {
...
}

⇒ Works!

Important Note: JavaScript only hoists declarations, not initializations. The variable will be undefined until the line where its initialized is reached.

Hoisting with let and var
With let and var hoisting is different

console.log(num) → Error if let or const
let num = 6; → with var undefined is printed

function expressions and class expressions are not hoisted