National University of Computer and Emerging Sciences



Web Programming

Es6

|  |  |
| --- | --- |
| Roll Number | 19F-0228 |
| Name | Muhammad Zain |
| INSTRUCTOR | Sir Shehroz khan |
| Semester | FALL 2021 |

ES6

What’s ES6? What’s the difference between ES5 and ES6? Write  
down any 5 new features introduced in ES6.

## ES6:

ES6 stands for **ECMAScript 6**. ECMAScript was created to standardize JavaScript, and ES6 is the 6th version of ECMAScript, it was published in 2015, and is also known as ECMAScript 2015

## ES5 & ES6:

ES5 stands for ECMAScript 5. It is also known as ECMAScript 2009. ECMA script is a trademarked scripting language specification defined by Ecma international. The fifth edition of the same is known as ES5ECMA script is a trademarked scripting language specification defined by Ecma international. The sixth edition of the same is known as ES6. Whereas ES6 stands for ECMAScript 6 (ECMAScript 2015).ES6 enables us to write a code that is more readable, concise, and efficient.

## New Features:

some of the best and most popular ES6 features that we can use in your everyday JavaScript coding.

1. let and const Keywords
2. Arrow Functions
3. Multi-line Strings
4. Default Parameters
5. Template Literals
6. Destructuring Assignment
7. Enhanced Object Literals
8. Promises
9. Classes
10. Modules

# Define let and const and explain how they differ from var. Explain Promises?

## let and const keywords :

The keyword "let" enables the users to define variables and on the other hand, "const" enables the users to define constants. Variables were previously declared using "var" which had function scope and were hoisted to the top. It means that a variable can be used before declaration. But, the "let" variables and constants have block scope which is surrounded by curly-braces "{}" and cannot be used before declaration.

let i = 10;

console.log(i); //Output 10

const PI = 3.14;

console.log(PI); //Output 3.14

## Promises:

A promise is an object which can be returned synchronously from an asynchronous function. It will be in one of 3 possible states:

1. **Fulfilled:** onFulfilled() will be called (e.g., resolve() was called)
2. **Rejected:** onRejected() will be called (e.g., reject() was called)
3. **Pending:** not yet fulfilled or rejected

A promise is **settled** if it’s not pending (it has been resolved or rejected). Sometimes people use *resolved* and *settled* to mean the same thing: *not pending*.

Once settled, a promise can not be resettled. Calling resolve() or reject() again will have no effect. The immutability of a settled promise is an important feature.

Native JavaScript promises don’t expose promise states. Instead, you’re expected to treat the promise as a black box. Only the function responsible for creating the promise will have knowledge of the promise status, or access to resolve or reject.

Acknowledgements:

Google.com

<https://www.geeksforgeeks.org/difference-between-es5-and-es6/>

<https://www.codesdope.com/blog/article/javascript-es5-vs-es6/>

<https://www.boardinfinity.com/blog/top-10-features-of-es6/>

# <https://medium.com/javascript-scene/master-the-javascript-interview-what-is-a-promise-27fc71e77261>:

# Q3 in folder: