JavaScript Essentials

Day 1 Assignment

Question 1: Explore and explain the various methods in console function.

Answer: a) console.log() : Mainly used to log(print) the output to the console. We can put any type inside the log(), be it a string, array, object, boolean etc.

b) console.error() : Used to log error message to the console. Useful in testing of code. By default the error message will be highlighted with red color.

c) console.warn() : Used to log warning message to the console. By default the warning message will be highlighted with yellow color.

d) console.clear() : Used to clear the console. The console will be cleared, in case of Chrome a simple overlayed text will be printed like : 'Console was cleared' while in firefox no message is returned.

e) console.time() and console.timeEnd() : Whenever we want to know the amount of time spend by a block or a function, we can make use of the time() and timeEnd() methods provided by the javascript console object. They take a label which must be same, and the code inside can be anything(function, object, simple console).

f) console.table() : This method allows us to generate a table inside a console. The input must be an array or an object which will be shown as a table.

g) console.count() : This method is used to count the number that the function hit by this counting method.

Question 2 : Write the difference between var, let and const with code examples.

Answer:

Var : The JavaScript variables statement is used to declare a variable and, optionally, we can initialize the value of that variable.

Example: var a =10;

Variable declarations are processed before the execution of the code.

The scope of a JavaScript variable declared with var is its current execution context.

The scope of a JavaScript variable declared outside the function is global.

Consider the following code snippet.

```
function nodeSimplified(){
  var a =10;
  console.log(a);  // output 10
  if(true){
   var a=20;
   console.log(a); // output 20
  }
  console.log(a);  // output 20
}
```

In the above code, you can find, when the variable is updated inside the if loop, that the value of variable "a" updated 20 globally, hence outside the if loop the value persists. It is similar to the Global variable present in other languages. But, be sure to use this functionality with great care because there is the possibility of overriding an existing value.

let

The let statement declares a local variable in a block scope. It is similar to var, in that we can optionally initialize the variable.

Example: let a =10;

The let statement allows you to create a variable with the scope limited to the block on which it is used.

It is similar to the variable we declare in other languages like Java, .NET, etc.

Consider the following code snippet.

```
function nodeSimplified(){
  let a =10;
  console.log(a);  // output 10
  if(true){
   let a=20;
   console.log(a); // output 20
  }
  console.log(a);  // output 10
}
```

It is almost the same behavior we see in most language.

```
function nodeSimplified(){
```

```
  let a =10;
  let a =20; //throws syntax error
  console.log(a);
}
```

Error Message: Uncaught SyntaxError: Identifier 'a' has already been declared.

However, with var, it works fine.

```
function nodeSimplified(){
  var a =10;
  var a =20;
  console.log(a);  //output 20
}
```

The scope will be well maintained with a let statement and when using an inner function the let statement makes your code clean and clear.

const

const statement values can be assigned once and they cannot be reassigned. The scope of const statement works similar to let statements.

Example: const a =10;

```
function nodeSimplified(){
  const MY_VARIABLE =10;
  console.log(MY_VARIABLE);  //output 10
}
```

As per usual, naming standards dictated that we declare the const variable in capital letters. const a =10 will work the same way as the code given above. Naming standards should be followed to maintain the code for the long run.

Question 3: Write a brief intro on available data types in JavaScript.

Answer: JavaScript provides different data types to hold different types of values. There are two types of data types in JavaScript:

Primitive data type

Non-primitive (reference) data type

JavaScript is a dynamic type language, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine.

There are five types of primitive data types in JavaScript. They are as follows:

a) String: represents sequence of characters e.g. "hello"

b) Number: represents numeric values e.g. 100

c) Boolean: represents boolean value either false or true

d) Undefined: represents undefined value

e) Null: represents null i.e. no value at all

The non-primitive data types are as follows:

Object: represents instance through which we can access members

Array: represents group of similar values

RegExp: represents regular expression