JS Interview Prep

**Q. Define box model?**

A. Every HTML element is considered to be wrapped inside a box that consists of margin, border, padding and the actual content.

**Q. What is CSS Selector?**

A. CSS selectors are like element identifiers that you can use to find or select the HTML element that you want to style.

**Q. Difference between display none and visibility hidden?**

A. Display none hides the html element from the viewport and the proceeding element takes its place whereas visibility hidden, hides the element but the element occupies its space.

**Q. What is flexbox in CSS?**

A. Its a one dimensional layout model. Flex allows responsive elements within a container to be automatically arranged depending upon screen size. It is useful in aligning the elements and distributing the spaces between them.

**Q. Diff b/w flexbox and CSS grid?**

CSS grid is a 2-dimensional layout model as it can handle both columns and rows whereas flexbox is one dimensional layout model, it can arrange the elements either in row or column.

**Q. JS is interpreted or compiled?**

A. JS is said to be a single threaded interpreted language however this is not entirely correct as some JS engines provide just in time compilation.

**Q. What do you mean by JIT?**

A. JIT compilation is compiling the bytecode to machine code at run time. For example, V8 JS engine does not compile the JS code before execution rather it does it while executing the code so the interpretation and the compilation goes hand in hand.

**Q. JS is single-threaded meaning?**

A. Yes, JS engines have one call stack that manages the execution context where the code is executed line by line, one piece at a time.

**Q. Diff between undefined and not-defined?**

A. When something has been defined in a scope but it's value is not provided it will have a default value which is undefined.

When something has not even been defined that is when it will throw a reference error saying that this variable is not defined.

**Q. How does hoisting actually works?**

A. JS program runs in two phases,

i) Creation of execution context within the call stack and assigning memory space to all the variables with values as undefined and functions as their declaration.

ii) Line by line code execution.

**Q. Do let variables get hoisted?**

A. Yes they are hoisted but they get into TDZ or (Temporal Dead zone ) which means they are inaccessible till the code execution reaches the line where they have been declared and any attempt of accessing these variables till they are in TDZ will throw a reference error.

=> Const variables are needed to be assigned a value at the time of initialization and will throw a syntax error if not done so.

**Q. Why do we call JS dynamic language?**

=> We call JS as a dynamic language coz the datatype of the variables can change at run-time.

for ex:=

let a = true //currently a is of type boolean

a = "true" // now the type of a is a string

so on and so forth...

**Q. How does JS determine data type?**

=> JS look at the type of the value that is assigned to the variable

so if it's say 100 type will be number if it's "Hello" type will be String so on and so forth....

**Q. What is typeOf operator or how can we determine type of a variable in JS?**

=> typeOf operator helps us in getting the datatype of any variable that is supplied to it and with help of this typeOf operator we can determine the data-type.

**Q. What are different data-types in JS?**

=> Major ones are SNNUBO

String Null Number Undefined Boolean Object other two are big int and Symbol.

**Q. Explain undefined dataType?**

=> undefined is the default data type given to every variable when it is declared but not assigned or initialized with a value. And it generally defines the absence of data.

**Q. Expain Null or diff b/w undefined and null?**

=> Null is same as undefined as far as the value prospect goes but the key difference is it is the intentional absence of data so i can say.

for ex:=>

let a; // absence of data but undefined

let b = null; // intentional absence of data and null

**Q. Hoisting?**

=> The javaScript program runs in two phases:

i) Creation of execution context within the call stack and assigning memory space to all the variables with value as undefined and functions as their declaration.

ii) line by line execution

Hoisting is a mechanism where variables and function declaration are moved to the top of the scope.

**Q. Are javaScript initialization hoisted?**

=> No only the declarations are hoisted.

**Q. What are global variables?**

=> Global variables are those which are available/ visible throughout the context of the program. That is can be accessed anywhere in the program.

**Q. What are the issues with Global variables ?**

=> It can make an application very hard to debug and buggy

**Q. What happens when you declare a variable with out var?**

=> The variable becomes global.

**Q. What is the use of "use strict"?**

=> "use strict" checks if the variable is declared with var keyword or not and if not it throws an exception of Reference error.

**Q. How can we avoid global variable issues?**

=> We can put them in namespace. Or we can use Module with closures.

**Q. What is closures in javaScript?**

=> Closures are function inside a function and they make functions stateful functions

**Q. Why do we need closures?**

=> To create Self contained modules. results in self contained stated and in avoiding global variables issues. It helps you to expose only what you want

**Q. Explain IIFE?**

=> Immediately invoked function expression, these are anonymous functions which invoke or call themselves immediately.

**Q. What is the use of IIFE?**

=> It solves the name collision problems. Let's say we have one function namd Init() but later on we defined a variable as Init. We can use an anonymous function and immediately invoke it to solve this problem.

**Q. What is name collision in global scope?**

=> Name collision happens when same name function names and variable names are declared in same context.

**Q. What are design patterns?**

=> Design patterns are time tested solutions.

**Q. What is the most used design pattern in JS?**

=> Module design pattern on the Module revealing pattern.

**Q. What is module revealing pattern?**

=> Module pattern or revealing module pattern has 2 big advantages:

-- Self-contained independent components.

-- Provides Encapsulation and Abstraction.

Module revealing pattern is a combination of IIFE and closures

=> better code management because of IIFE and good encapsulation because of closures.

**Q. IIFE vs normal function?**

=> A normal function simply inclueds a declaration and a definition but na IIFE is declaration, definition and invocation at the same time.

**Q. What are the different ways of declaring objects in js?**

=>

i) Using literals that means {key: value} giving a key value pair.

ii) Using Object.create();

iii) Constructor way that means by using functions

iv) ES6 class method

**Q. How can we do inheritance in JS/ What is protoype in JS/ Explain Prototype chaining?**

=> JS uses object inheritance or prototypical inheritance. Inheritance is done using prototype object.

for ex:= Manager.prototype = emp // Manager is a constructor and emp is an object of another constructor.

Prototype chaining is a process where the property methods are first checked in the current object, if not found then it checks in the prototype object, if does not find in that it try checking the prototypes prototype object, until it gets the prototype object null.

**Q. What is let keyword in js?**

=> Let is also an ES6 feature and it helps to create immediate block level local scope variables.

let variables do get hoisted but they get into TDZ Temporal dead zone and they do not get initialized even with an undefined.

if tried to access

Uncaught ReferenceError: Cannot access 'x' before initialization

**Q. What is temporal dead zone?**

=> It's a period or it's a state of a variable where variables are named in memory but they are not initialized with any value.

**Q. let vs var?**

=> 2 differences:

i) Scoping rules: var are scoped to immediate function body, whereas let are scoped to immediate enclosing block.

ii) Initialized value during the hoisting phase: var are initialized with undefined, let are initialized with nothing

**Q. String concatenation and arithmetic puzzle?**

=> "10"+"10" = "1010"

10+10 = 20

1 + 1 + "4" = "24"