Topic:- JS introduction
Author:- Sudhendra Kumar Singh

Topics to discuss:-

1. JS Introduction
2. Scripting language vs programming language.
3. var, let, and const
4. Chrome Devtools
5. Type Coercion:- https://javascript.info/type-conversions
6. Arithmetic (short-circuiting)
7. Functions
    a. No return type
    b. Missing parameter
    c. No param type
8. Function hoisting
9. Function overloading - https://youtu.be/tYORbcT1OVQ  // to do later
10. Functions and scope
    a. Global
    b. Function / lexical
    c. Execution context
        i. Variables and environment
        ii. This
        iii. Reference to outer execution contexts (not for global context)
11. Function expression (assigning function to a variable)
    a. Named function
    b. Anonymous function
12. Types of errors javascript:- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Errors
13. Function as a parameter & nested function
14. Window prompt, alert, and confirm.
15. Typescript vs Javascript // discuss later
16. Arrays in javascript
    a. Can assign value to any index even if we have no elements in between.
    b. Arr.length
17. Functions on arrays
    a. Push
    b. Pop
    c. Shift
    d. unshift
    e. Splice
    f. Slice, …..read other on MDN docs
18. Iterating over the array
    a. Foreach
    b. Map
    c. For
    d. For-in

    e. Foreach and map difference.
    f. for-of
19. Objects in javascript
    a. JS is an OOP language
    b. Key-value pair
    c. Add property
    d. Square brackets -> used when keys are not valid as per JS rules.
    e. Heap storage // till here
20. Clone objects
    a. Spread -> works on iterable entities
    b. Assign
    c. Parse and stringify
    d. let clone = Object.create(Object.getPrototypeOf(obj), Object.getOwnPropertyDescriptors(obj));
21. Cloning of nested objects // do it later
22. Iterate over object
    a. Objects.keys
    b. Object.getownpropertynames
    c. For in loop
    d. Object.entries
    e. Difference b/w keys / getownpropertynames // do it later
23. Array as an object:-
24. typeof typeof 1
25. Array length is max integral index
26. Callback functions
27. Timing functions
    a. Settimeout
    b. Setinterval
    c. Setimmidiate //try out in HTML
28. DOM
    a. documentElelment
    b. Head
    c. Body
    d. Window.screen
    e. Window.location
29. List, nodelist, map, set, arrays
30. API
31. Lamdba function // discuss later
32. Accessing DOM
33. Element handle
34. Getting element :-
    a. Getele...id
    b. Getel….tag
    c. Getel… class
    d. Queryselector
    e. Queryselectorall
35. Event handling
    a. Onclick / addlistener

36. Script tag and js code
37. Handle multiple triggers
38. Mouse and keyboard events
    a. mouseover
    b. mouseout
    c. Keypress  (only some keys are detected, alphabet, number)
    d. Keydown (all down)
    e. Keyup
    f. event.keyCode
39. Event Propagation :- https://javascript.info/bubbling-and-capturing
    a. Stoppropogation
    b. Strict mode
    c. Strict mode in a particular scope
40. Hosting file on GitHub (normal html) //do it later
41. Navigator - your browser identity
42. matchMedia  //MDN docs
43. Quizzes on DOM and timing events
44. Scopes revision
45. What happens when we type the address and hit enter // do it later.
46. Single thread/multithread
47. Avoiding global variables
48. IIFE -> Immediately invoked function expression
    a. Ways to create IIFE:- https://javascript.info/var
49. let, const, and var
50. {}  -> block and solving same variable problem
51. Closures  (function + lexical environment in which it was created)
52. Closures -> preserving scope
53. Closures and let
54. var in a for loop
55. Arrow functions
56. Binding in arrow functions  // done till the last session
57. This keyword: - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this
58. This in strict mode - read MDN docs
59. To change function this make the function a property of that object.
60. Functions to create objects
61. Call, apply and bind
62. New keyword binds to new object; without new keyword, ‘this’ refers to upper scope.
63. Every time two objects are created when create a function
    a. Object
    b. It’s prototype
64. Constructors
65. Prototype
    a. Share member function, data members, and properties among objects.
    b. Objects have access (have reference) to their prototype.
    c. Add property to prototype at runtime. (in Javascript)
    d. Similar to static members' behaviour in other languages.
66. Accessing prototype from an object:-

    a. obj.__proto__  (dunder proto)
    b. Object.getProtoTypeOf(obj)
    c. Hasownproperty:- to check its property or prototype's property
    d. Both objects and their prototype can have the same property.
67. Object as a function
68. Prototype chaining
    a. Every prototype has a reference to Object.prototype
    b. Lookup happen up the chain
69. Browser storage API (to do later)
70. Object and object
    a. Object :- constructor function
    b. object:-  non-primitive data type (key-value pairs)
71. Difference function and class
    a. Can't call a class without a new keyword
    b. The class declaration is not hoisted
72. Every object has reference to its prototype [[Prototype]]  (Object inherit from prototype)
73. Class and function expressions are not hoisted
74. Inheritance in class and super keyword.
75. Javascript is synchronous.
76. Asynchronous
77. Event queue / Event loop
78. Synchronous and asynchronous code
79. Client Server architecture
80. Ajax -> Asynchronous javascript and XML (uses HTTP to transfer data)
81. XMLHTTPRequest
82. Method:-
    a. Get -> fetching data from the server.
    b. Post -> client is sending some data to the server.
    c. Put -> update an existing data/entity.
    d. Patch -> part of data updated.
    e. Delete -> delete an entity/data.

    More info on Rest API:- https://www.redhat.com/en/topics/api/what-is-a-rest-api
83. JSON
84. Read about JSON
85. HTTP  (protocols are specified rules)
86. Steps for a client-server connection
    a. Client setups a new connection with the server.
    b. The client makes a request to server.
    c. Server receives the request and processes the request.
    d. Server sends back the response
    e. Client closes the connection.
87. Types of request
88. Fetch API
89. Response codes  (MDN docs)
90. HTTP is stateless protocol
91. API (browser API -> local storage, cookies, media) // discuss later
92. CORS (Cross-Origin Resource Sharing)

93. Hitting a public API
94. JSON parse / JSON stringify
95. Authenticated API
96. https://api.nasa.gov/
97. Promises
    a. Pending
    b. Settled
        i. Fulfilled
        ii. rejected
98. Promise methods (all, race, allsettled, any, resolve, reject) //interview question
99. Call back function
100. Javascript is a single-threaded language that handles async tasks with a callback.
101. Then and catch, finally (asynchronous)
102. Call back hell / Pyramid of Doom
103. Passing variable inside a callback.
104. Javascript event loop
    a. Stack
    b. Queue
    c. Heap
    d. Message
    e. Frame
    f. non-blocking
105. Example :- 0 delay setTimeout
106. Promises and Microtask queues (order of execution with call stack and macrotask queue)
107. Arguments array
108. Git
    a. Why git
    b. Installing git
    c. Basic commands:- git init, git add, git status, git commit, git merge.
    d. Reference :- https://wac-cdn.atlassian.com/dam/jcr:e7e22f25-bba2-4ef1-a197-53f46b6df4a5/SWTM-2088_Atlassian-Git-Cheatsheet.pdf?cdnVersion=77
109. Try out https://javascript.info/
110. Quizzes :-
    a. https://github.com/lydiahallie/javascript-questions
    b. https://github.com/sudheerj/javascript-interview-questions
111. Note:- Wrap object in small bracket before returning from arrow function.
112. DNS
113. Projects:-
    a. Ping-pong game
    b. Movie app
114. Extras:-
    a. Unhandledrejection event in window object
    b. Set /Map
    c. Generator functions
    d. Symbol

e. Debounce and throttle

115. Objects that can be used in for….of are called iterables
116. http://es6-features.org/#Constants - ES6 features
117. if (true) let a = 1; // this won't work, syntax error.
118. Temporal dead zome based on time of execution

## Interview Questions

```
---------- JAVASCRIPT -----------
Basic
1. console.log( typeof typeof 1 )
2. var, let, const
3. forEach(), map(), filter(), find()
4. slice() vs splice()
5. WOM (window object module)

Intermediate
1. console.log( typeof null )
2. Object.assign()
3. ({ a } = b)
4. Error types // Reference error, assertion error, syntax error, range error etc.
5. Callback hell, deadlock, event loop
6. High order function vs callback function

Advanced
1. hoisting
2. bigint // datatype for >= 2^53 && <= -2^53
3. number datatypes that are word -> Infinity, +Infinity, -Infinity, NaN
4. what is e in number // 3e5 = 3 * 10^5
5. V8 engine (chrome)

----------- REACT ---------------
Basic
1. react lifecycle
2. state vs props

Intermediate
1. Pure components
2. Higher order components
3. Context API and useReducer()

Advance
1. React.lazy()
2. Error boundaries

-------------- CSS ----------------
Basic
1. Box model // why outline not comes in box model
2. Flex and grid
3. box-sizing
```

```
Intermediate
1. Visibility hidden vs display none vs opacity zero
2. suedo classes vs suedo elements
3. % vs vh/vw
4. @fontface
5. keyframes

Advanced
1. min(), max(), minmax(), clamp()
2. backface-visibility
3. white-space
4. perespective
5. box-decoration

------------- HTML --------------
Basic
1. inline elements and block elements
2. id vs classes
3. symentics

Intermediate
1. meta tag
2. font icons vs svg
3. alt attribute in <img>

Advance
1. data attribute and aria attribute
2. canvas
3. <picture>


// Fullstack preparation topics
```

https://expressjs.com/en/5x/api.html

mongoose

You

20:20

passport

chatsocket

nodemailer

express

http, express

jwt

You

20:21

React Topics:-

hooks

react-redux

lifecycle methods

You

20:22

useEffect, useState

https://react.dev/reference/react/useCallback

You

20:24

Old React documentation:-

https://legacy.reactjs.org/docs/hello-world.html

react router

You

20:26

https://docs.google.com/document/d/1eD4bm6i4hl5vmxuN0js8goP2aCODaZunS8zeHuoHo50/edit