Practice Interview/QC

1. **Previous Week’s**
2. What do you know of Action methods?
3. What is DNS?
4. Why do we need Web Services?
5. What are control flows in JavaScript?
   1. Type coercion normally occurs during control flows.
6. Reflect on JavaScipt, CSS, and HTML. How do these (functionally) relate to each other/How do these interact with each other on the web?
7. What exactly is loosely coupling?
8. Compare and Contrast AJAX and the Fetch API?
   1. They both help us make our API calls.
   2. AJAX does it by opening an XHR object that is then used to open the request & must capture that data in the XHR object.
      1. AJAX requests goes through 4 different states before coming back to us in JSON format.
      2. Thus, AJAX are unable to implement Promises
   3. Fetch implements the use of Promises & that’s a way of making a callback function in our requests.
      1. Promises makes it easier to implement.
9. What is the contract for a SOAP service?
10. How is a SOA architecture different from a monolithic architecture?
11. What are some best practices for making our web services?
    1. Web services are for presenting info to a user or another service – so make the info receivable & usable
    2. Ensure there’s clear documentation for the resources that are being accessed for clients to understand & match their expectation.
    3. Ensure on the backend that things are structured to help the functionality of the greater service
12. Run me through the AJAX lifecycle.
13. Would concurrency be an issue with AJAX?
    1. Concurrency is when 2 operations occur in your program at once.
    2. It wouldn’t be an issue because it would execute and give you the results based on the order that they finish.
14. What are all the HTTP methods?
15. What is the makeup of an HTTP request?
16. What HTTP methods are safe verbs?
17. What HTTP methods are idempotent?
18. Is CORS a protocol?
19. When you have some arrays in your JavaScript code, how would you access an element in your array?
20. What is type coercion in JavaScript?
21. What are the function types in JavaScript?
22. What are the different scopes in JavaScript?
23. What are the types of web services?
24. What is the difference between undefined & null?
25. What is faster: SOAP or REST?
26. What is the messaging format of SOAP?
27. What protocols does SOAP use?
28. What protocols does REST API use?
29. What is more secure: SOAP or REST API?
30. How can you create a SOAP service in .NET?
    1. Is there a specific framework to create a SOAP in .NET?
31. How can you create a REST API in .NET?
    1. Is there a specific framework to create a REST API in .NET?
       1. Web API framework
32. How is the WSDL used in REST API?
33. How is the WSDL used in SOAP?
    1. WSDL offers more security on top of itself where data cant be manipulated between transfer
34. How does SOA better structure our APIs?
35. What is SOA all about?
36. API controller vs MVC controllers?
    1. MVC controller returns views because it is inherited from Controller class
    2. API Controller returns data because it is inherited from API Controller
    3. Thus because of the different inheritance they have different action methods
37. What are the drawbacks of SOA?
    1. Can you give an example of a drawback of SOA and an alternate architectural pattern?
38. What are the RESTful constraints/principles?
39. Can you explain HATEOAS?
40. What is a resource in a RESTful service?
41. What is Lexington scope?
    1. There is a global scope & a block scope in JS.
    2. Lexington scope is just a special block scope.
       1. How to designate between Lexington scope & function scope is by using the let keyword.
42. Can a datable, like records be accessed/called as a resource?
    1. What about a tomato? -Is the tomato is an object (Answer)
43. What is content negotiation in RESTful API?
    1. Keeping track of what you’re getting from a database
    2. Where can I request this?
44. What is model binding in ASP.NET Core?
45. What are the different ActionResults in a web API?
46. How would you do error handling in ASP.NET Core API?
    1. Use a catch statement to ‘catch’ & can also ‘catch in a then statement
47. With ES6, what are some of the new features in JavaScript?
    1. Let, the ability to use asynchronous methods, try/catch blocks?
    2. What is this asynchronous function then?
       1. The use of an async modifier & the implementation of await
48. What makes something asynchronous?
    1. You have to wait for a function to finish (a different function must complete a complete a task and return)
49. What properties make JavaScript usable/viable/useful as a programming language?
    1. JIT, easy to deploy, can be
50. What are the drawbacks of JavaScript?
    1. Debugging is harder due to JIT compilation
51. What are some paradigms that JavaScript supports?
    1. OOP, imperative, declarative (functional programming) paradigms?
52. What is the WSDL (wiz-dol)?
53. **Previous Weeks’ Content**
54. What is AJAX?
    1. Can you explain how AJAX works/the AJAX lifecycle?
    2. What are the two major features of AJAX?
    3. What is the difference between AJAX & Fetch?
55. What is a web service? Web services are XML-based, asynchronous applications online that are invoked using Remote Procedure Calls (RPC).
    1. RPC is basically asking a remote server to do something for you
    2. Webservices have:
       1. SOAP – way of communicating
       2. WSDL – Describes to clients what the service does, info to connect, and function/methods, and location
       3. UDDL – a repo for WSDL files to be published by devs
    3. Give some examples of a web service.
    4. Compare and contrast a webservice and a non-web service.
56. What is SOAP?
    1. What has replaced SOAP?
    2. SOAP uses FTP, STP, and HTTP
    3. What are the advantages and disadvantages of SOAP?
       1. Firewalling
       2. Marshalling & unmarshalling
    4. Compare and Contrast SOAP and REST APIs.
       1. REST allows you to do CRUD but SOAP has preset.
       2. SOAP is strictly XML
       3. REST can use JSON and text
    5. What is WCF?
       1. Microsoft proprietary framework that enables use of SOAP in .NET
57. What is SOA/Service oriented Architecture?
    1. What are the primary uses of SOA?
    2. What are the characteristics of SOA?
    3. What are the benefits of SOA? What are the limitations of SOA?
58. What is CORS? CORS stands for cross-origin resource sharing and allows a website from one url to access resources from another URL. However, it is not enabled by default because many browsers implement the same origin policy by default, meaning only resources from the same origin URL can be accessed.
    1. What are CORS standard protocols?
    2. How is CORS implemented?
       1. If a client sends a request to a server not-the-origin, the server will add the access-control-allow-origin-header in the response. If the value matches the origin (or is wildcard) then the client can access the resource, but otherwise cannot.
       2. If you don’t control the server, there’s not much you can do asides from ask the managers of the server. If you control the server, you can implement CORS for a client’s origin by
       3. On a side note, these extra headers are what allow clients to make only specific types of requests. For example, the access control allow methods header states which http methods are allowed to go through
       4. Preflighting is like screening somebody to go onto an airport – you’re checking to ensure somebody is safe to go on a flight, bu instead of checking luggage you’re checking their origin and the method they are sending with the request.
       5. How is CORS implemented in .NET?
          1. You have to add CORS in the Configure services method and create a policy accordingly. Then you implement app.userCORS line and the CORS policy you created earlier AFTER userouting and BEFORE useauthentication
    3. What is CSRF?
       1. Cross-Site Request Forgery is a type of attack where unauthorized commands are transmitted from a user that the web application trusts.
    4. Give an example of CORS.
    5. Give an example of a client or API that wouldn’t need CORS.
59. What is Fetch API?
    1. How do we use fetch API?
    2. What response does fetch return?
       1. How can these responses be handled?
    3. What datatypes can be returned from fetch responses?
    4. What is the main benefit of the Fetch API?
60. What is a REST API/What are the guiding principles of REST? Representational State Transer, aka REST, is an architectural approach to designing web services, where data and functionality are resources that can be accessed through a Uniform Resource Identifier, and a standardized protocol HTTP.
    1. How does REST utilize HTTP methods?
       1. What are some examples of HTTP methods?
       2. Briefly explain the HTTP lifecycle
    2. What is the Richardson Maturity Model?
       1. A model that allows us to know how RESTful an API is. Every REST API belongs to one of the 3 levels (and non-restful API’s are level 0).
    3. What is the “holy grail” of REST and how can it be achieved?
       1. With HATEOAS, a client interacts with a network application whose application servers provide information dynamically through hypermedia.
       2. HAETOAS is when a client interacts with a network application whose server includes dynamic hypermedia to resources in the response to the request.
    4. What is filtering?
       1. When you apply a filter so that only certain data values are retrieved
    5. What is pagination?
       1. When you specify only a certain number of items to return with a request at a time, and retrieve next set of X size when user accesses new page
    6. What is OAI?
       1. Can you give some examples of OAI members/interfaces?
    7. What are the guiding principles of RESTful API? CCCLUS
       1. Code on demand (Allows clients to have increased functionality by downloading code scripts or applets)
       2. Uniform interface (All interfaces have 4 constraints: ID of resources, descriptive messages, HATEOAS)
       3. Cacheable (data can be temporarily saved onto client)
       4. Client-server (separates user interface from data storage concerns)
       5. Layered system (Each component can’t see beyond the layer it’s interacting with)
       6. Stateless (each request must be independent of other requests & contain all information to understand the request)
61. What is JavaScript? lightweight, [just-in-time](https://en.wikipedia.org/wiki/Just-in-time_compilation) compiled programming language with functions. It is most well-known as the scripting language for Web pages. It supports object oriented style because of its classes & objects
    1. Is JavaScript OOP?
    2. How do variables function in JS?
       1. What are the different variable types?
    3. List all the JavaScript datatypes.
    4. List some important operands.
       1. Does JavaScript have any special order of operations rules?
          1. Answer: operator precedence
    5. Explain truthy and falsy.
    6. Explain the different equivalency operators.
    7. Explain type conversion in JavaScript.
    8. How does javascript JIT compilation work
    9. How do classes function in JS?
    10. How do objects function in JS?
    11. What is the main difference between classes & objects in JS, compared to OOP languages?
    12. What are JavaScript prototypes?
    13. What are anonymous functions
62. What is the DOM? (The Document Object Model which is an interface that represents a document’s content as nodes and objects so that a program can manipulate content and elements as necessary).
    1. What is the terminology for manipulating the DOM and how can one do so?
       1. Walking the DOM means using the DOM’s nodes to access and manipulate elements
    2. What is the primary useful feature of the DOM?
    3. What are selectors?
    4. What are events?
       1. What is bubbling?
       2. What is capturing?
    5. How are DOM events loaded ?
       1. DOMcontentLoaded event fires when the entire html content has been loaded and parsed, without styling or images finishing loading.
          1. THIS IS NOT THE SAME EVENT AS load, which only occurs once a webpage is FULLY LOADED.
       2. Events are objects based on the Event interface that notify code of changes that may affect code execution, such as key board presses, mouse clicks, form submission, and more.
       3. Events are logged by an Event Listener and the .addEventListener() helper function takes in the event, and a callback function to execute as arguments
       4. Events are handled an Event Handler, which is a block of code that says what to do after an event is fired
63. **WEEK 7**
64. What kind of checking does TS have?/How do I check in TS?
65. How does TS compile?
    1. What(/How many) kinds of compiling are there?
66. Why do we use TS?
67. When would I use TS over JS?
68. How does TS enforce strong typing?
69. What is structural typing?
70. What paradigms does TS support?
71. What are erased types?
72. What kind of datatypes does TS have?
73. How does TS support/implement inheritance?
74. Explain how classes operate in TS.
75. Explain how interface operate in TS.
76. When would I use an interface over a class in TS?
77. How do functions operate in TS?
78. What are TS modules?
79. What is Angular CLI?/What does Angular need to run?
80. What are Angular components?
81. What is the makeup of a component?
82. How is Angular structured to display information?
83. What are classes in Angular?
84. What are interfaces in Angular?
85. Why do we use DI in Angular?/How can Angular consume services?
86. What are forms in Angular?
87. How do forms consume information?
88. What is Reactive programming?