Benjamin Zhuang - Week 2 Day 3

1. What are the disadvantages of synchronous code?
    a. Synchronous code will **block**. If there were a block of code that's going to be executed for an indeterminate amount of time (i.e. fetching data from a web server, sending data, timeouts), the user won't be able to do most of the interactions on a webpage, such as scroll or click.
2. What is asynchronous code in JavaScript?
    a. Asynchronous code always executes after synchronous code. They will never block so that other synchronous code can be run.
3. Since JS is single-threaded, how does it achieve asynchronous code?
    a. It uses the event loop to handle the execution of synchronous and asynchronous code, which is implemented by a call stack and a task queue, or more precisely, the macro & micro task queue.
4. What does the event loop do? What data structures does it use?
    a. It handles the order of the execution of synchronous and asynchronous code. It uses two main data structures to implement the event loop, which are a call stack and a message (task) queue.
5. What is the callback queue?
    a. A callback queue is a data structure used by JavaScript to implement the event loop. And it is a data structure to process messages in order. That is, execute the oldest message from the callback queue first.
6. What is an HTTP request and HTTP response?
    a. An HTTP request is a way for the client to communicate with a remote web server. It's used to retrieve or submit some data to the server.
    b. An HTTP response is an object sent back by the web server upon an HTTP request as a way to notify the client that request has been received.
7. How many HTTP methods are there? Explain each one.
    a. There are many HTTP methods out there, The more common ones include,
        - GET.
        - POST.
        - PUT.
8. What is the difference between GET and POST? What about POST and PUT?
    a. GET only contains headers and sends additional data through the URI. In contrast, POST contains both headers and body and sends data through the body. If you want to send some sensitive information, such as username and password, you should use a POST request.

    b. PUT request only contains headers and it's idempotent, which means that it will not have side effects if multiple identical PUT requests were sent. On the other hand, POST requests are not idempotent.

9. Could you explain the different classes of HTTP status codes? What are some common status codes?
    a. 2xx (success)
    b. 3xx (redirect)
    c. 4xx (client side error)
    d. 5xx (server side error)
    e. Common status code includes 200 (success), 301 (permanent redirect), 302 (temporary redirect), 404 (page not found), 500 (internal server error)
10. What is AJAX?
    a. It stands for Asynchronous JavaScript and XML. It's a technique to use asynchronous code to communicate with servers, exchange data, and update the webpage without reloading.
11. What is XHR?
    a. It stands for XmlHttpRequest, which is an object that is used to perform AJAX requests to a web server. For ES6, a more commonly used library for making AJAX calls is the fetch API.
12. What is a Promise?
    a. A promise is an object that has 3 states – pending, fulfilled, rejected to represent an asynchronous operation.
13. How many states does a Promise have? What are they?
    a. 3. Pending, resolved, rejected.
14. What is callback hell?
    a. Nesting a bunch of callback functions whose argument depends on the outer callbacks. The code is really hard to debug and read with this syntax.
15. What is the advantage of Promises over callbacks?
    a. Promises can handle asynchronous operations more easily and you get better error handling with promises.
16. Explain Promise.all() vs Promise.allSettled().
    a. Promise.all() takes in a list of promises as the argument. If any of the promises is rejected, all promises will be rejected. A single promise object will be returned in this case, and it will be the first one to be rejected in the list of arguments. This can be useful if you have several promises that are related and need each other to resolve in order to continue execution.
    b. Promise.allSettled always resolves to an iterable, for which each entry represents their status & results.

17. What is the Microtask Queue?
    a. Microtask Queue is a data structure implemented by JS to achieve asynchronous code. The micro task queue has a higher priority than a macro task queue.
18. What is the difference between making server requests via fetch and XHR?
    a. There are many differences, but the main ones are that you have to check the status property for errors and chaining is difficult in XHR. The fetch API is more modern, it can be easily chained with .then() or async/await keyword. In addition, status errors can be caught by .catch() or try/catch blocks.
19. What is async & await? How do we use them?
    a. async / await are syntactic sugars for asynchronous programming that utilizes the Promise object and the promise chain. Async defines a function that must return a Promise object, and await is an unary operator, which can only be used in async functions, to halt the execution of the async function until the awaiting promise is resolved.
20. Explain localStorage vs sessionStorage.
    a. Both of them are memory storages, which are stored as key-value pairs according to the JSON format. localStorage would pertain even if the browser is closed unless the browser cache is explicitly cleared, sessionStorage would pertain only to the current tab. If it’s closed, data will be deleted.
21. What do the async and defer attributes do when loading scripts?
    a. async is downloaded in parallel. It loads the scripts with the possibility that the entire DOM tree is yet being fully constructed. It’s useful for some initialization code that doesn’t depend on the DOM elements.
    b. defer is downloaded in parallel. It loads the scripts after the entire HTML body is parsed. Useful if the script depends on previous scripts or requires full DOM.
22. What are ES6 modules? Why are they useful? How do we make these modules available in other JS files?
    a. A unit of code with related tasks and semantics that are used to modularize a large application framework.
    b. We use the import keyword at the top-level of modules to make them available.