DWA_01.3 Knowledge Check_DWA1

---

1. Why is it important to manage complexity in Software?

Complexity is an unavoidable aspect in programming and human error will always occur. When we manage complexity to the best of our ability we minimize the probability and the risk of a line of code in a sea of code to cause the whole system to break. In managing the complexity we also make it so that when something goes wrong it can be a smooth process to fix it.

---

2. What are the factors that create complexity in Software?

Composition: Complexity comes from composing hundreds/thousands of little functions together to become one overall system.
People: Lots of different people work on the same project/codebase over a long period of time.
Growth/Scaling: The codebase will inevitably grow bigger and bigger as more layers get added to the end product.
Programming/Alien material: The practice of coding itself is complex even after years of practice.
Technical Debt: Having a backlog of tasks that need to be done/code that needs to be cleaned up etc.

---

3. What are ways in which complexity can be managed in JavaScript?

Code Styling: Keeping a consistent code style can reduce noise and ultimately confusion
Documentation + Comments: Using JSDoc to make comments on not just how a certain something works but also what type of input/output it has is invaluable
Abstraction + Modularity: Using abstraction as if a piece of code is a software on it's own so that the less the person who is working with it needs to worry about it the better
Error Checks: Code in a way that the code can't run in the case of something getting mixed up or confused, throw errors
Using Object Oriented Programming + Functional Programming

---

4. Are there implications of not managing complexity on a small scale?

If I understand correctly, it depends on context. If you are building a small to do app, it's not crucial to implement strict practices to manage complexity unless you know it's going to scale. Although, I think using a coding style consistently should always be practiced to keep your code clean and as readable as possible.

---

5. List a couple of codified style guide rules, and explain them in detail.

From the Airbnb Style Guide:
Strings: If a string is longer than 100 characters, don't break it up into more lines. This is because broken strings are difficult to work with and make code less searchable.
Variables: Always declare variables using const or let to avoid making random global variables and always group together your const and let variables separately so if you need to assign a variable later on depending on a previously assigned variable it's easier to find/work with.
Comments: Always use /**...*/ for multiline comments and // for single line comments and always put the comments above the related code, this will help with consistent readability.
Comparison Operators: When doing an if statement you can use shortcuts for booleans but explicit comparisons for strings and numbers. I believe this is once again to reduce noise and keep things consistent in terms of readability. You know when you're working with a boolean and when not etc.

---

6. To date, what bug has taken you the longest to fix - why did it take so long?

In the capstone for Intro To Web, we had to deal with pagination for the first time. I had an object containing a list of books and I had to display 34 (I think) books per page. The problem came in when a user would search something different but the pagination still carried on. For example: If you were in the business category, and say you went over 2 pages, leaving you on page 102 and you would change over to a different category, let's say children's books... The children's books would start on page 102 instead of page 1 of that category. This took me a few hours to figure out and understand what the underlying issue is simply because it is logic that I haven't had to deal with yet. It felt a bit like a maze of logic and following the logic was difficult to wrap my head around, but eventually after logging things in the console and seeing exactly what is going on INSIDE of the code I realized all I needed to do was reset the page variable to page 1 every time someone pressed the search button to reset the pagination.
