DWA_01.3 Knowledge Check_DWA1

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

- For easier Maintenance of the software as it grows it becomes more challenging to understand and Modify.
- Better performance and Efficiency: complexity can have a negative impact on the software and makes it harder to optimize algorithms.Well managed code leads to better Utilization and faster execution.
- Reduced risk of errors: complex software is more open to opportunities for errors to occur.

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

- Human factors: Complexity is influenced By the experience,skills and Knowledge of the team. Inexperienced or understaffed teams, lack of communication, and inadequate collaboration can contribute to increased complexity.
- Size and scope: Large-scale systems with numerous features, functionalities, and interactions tend to be more complex than smaller projects.
- Rules and requirements of the Business: Systems that must handle numerous algorithms and perform a number of tasks that may be challenging to design and implement.

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

- Code commenting and documentation: Write comments in your code to provide clear explanations of complex logic and algorithms. Use comments to clarify intentions, assumptions, or any non-obvious behavior. Well-documented code improves readability and makes it easier for other developers to understand and maintain the code.
- Code organization and naming conventions: Make use of consistent and meaningful naming conventions for variables, functions, and classes.Clear and consistent naming helps in understanding the purpose and functionality of different components.
- Keep up with the times and emerging technologies: Stay up to date with the latest language features, and libraries in the JavaScript ecosystem. Following industry trends and frameworks can help simplify development and reduce complexity.

---

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

- Reduced maintainability: Without managing complexity, code can become difficult to understand. As a result, making changes or fixing issues becomes more time-consuming and error-prone. Over time, this can lead to a decrease in the maintainability of the software, making it harder to add new features or address bugs.
- Increased risk of errors: Complex and unmanaged codebases are more prone to errors and bugs. When code is difficult to understand, developers may unintentionally introduce mistakes during modifications or additions. These errors can damage the software's functionality, stability, and security.
- Wasted time and effort: Not managing complexity can lead to wasted time and effort during development and maintenance. Developers may spend excessive time solving code, tracking down issues, or working around tangled dependencies. This reduces productivity and may result in missed deadlines or incomplete features.

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

- Use descriptive variable and function names: Descriptive names for variables and functions enhance code readability and maintainability. Instead of using generic names, make use of names that accurately describe the purpose and functionality of the variable or function. This makes it easier for other developers to understand the code without needing to dive into the implementation details.

- Use consistent indentation and whitespace: Consistent indentation and whitespace formatting improve code readability and maintain a clean code style. Maintaining a consistent indentation style (e.g., using spaces) and spacing around operators and keywords helps make the code visually organized and easier to follow.

- Limit the length and complexity of functions and methods: Functions and methods should be kept focused on a single responsibility. Limiting the length and complexity of functions improves code readability, understandability, and maintainability. By maintaining this rule, it becomes easier to identify and isolate issues or modify specific parts of the code without impacting unrelated functionality.

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

It was on my final capstone project the bug was when the book review app showed the books I could not press any book to show its preview so there were other books that would show the preview. It took me so long to figure out because I did see that I had to loop the button maker so that it makes the books all buttons so that they can be clicked.