DWA_01.3 Knowledge Check_DWA1

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

Managing complexity in software is important for maintainability, readability, reduced error rates, scalability, cost savings, and delivering a better user experience. It allows developers to create software that is easier to understand, modify, and extend while ensuring high-quality, reliable, and efficient systems.

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

- Size and Scope -The larger the software system and the more features it has, the higher the complexity.
- Dependencies and External Factors - Software often relies on external libraries, frameworks, APIs, or services. Changes or updates to these external factors can introduce compatibility issues or unexpected behavior, increasing complexity.
- Lack of Documentation - When developers struggle to understand how different parts of the software interact or lack information about design decisions, it becomes more challenging to modify or extend the system.

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

Through maintainability - As software systems grow in size and complexity, it becomes harder to understand and modify them, breaking down the system into smaller, modular components with clear responsibilities and interfaces.

Readability and Understandability - Software is commonly developed by teams of developers, and often, different developers work on different parts of a system. So managing complexity ensures that code is readable and understandable, allowing for better collaboration, easier debugging, and new team members to adapt easily.

Scalable - . Managing complexity enables the design of scalable architectures that can handle growth without sacrificing performance.

---

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

By not managing complexity on a small scale, you risk creating a codebase that is difficult to maintain, prone to errors, challenging to scale, and costly to work with. Cautiously managing complexity from the early stages of development can help reduce these risks and lead to more efficient and sustainable software projects.

---

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

Rule: Use descriptive variable and function names - This rule emphasizes the importance of using meaningful and descriptive names for variables and functions. Instead of using generic names like "x" or "temp," developers should choose names that accurately describe the purpose and intent of the variable or function.

Rule: Use consistent indentation and formatting - This rule emphasizes maintaining consistent indentation and formatting throughout the codebase. Consistency in formatting makes the code more visually appealing, readable, and helps maintain a uniform coding style across the project.

---

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

Bugs that don’t log to the console - Tracing through numerous files, modules, or libraries to identify the source of the bug can be challenging.
