DWA_01.3 Knowledge Check_DWA1

---

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

Managing complexity in software is important because it enhances understandability, maintainability, and scalability, leading to more efficient development and easier collaboration among teams.

---

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

Factors that create complexity in software include inadequate design, poor documentation, tight coupling between components, lack of modularization, frequent changes in requirements, and the use of complex algorithms or data structures.

---

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

Complexity in JavaScript can be managed through various techniques such as modularization and encapsulation, following coding best practices like separation of concerns, using design patterns, employing libraries or frameworks for abstraction, writing clean and maintainable code, and conducting regular code reviews and refactoring.

---

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

Yes, there are implications of not managing complexity on a small scale in software development. It can lead to code that is difficult to understand, maintain, and debug. It increases the chances of introducing bugs and decreases the overall productivity of the development process. It also hinders code reuse and can make it harder to adapt or extend the software in the future.

---

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

1. Rule: Use meaningful and descriptive variable and function names.
   Explanation: This rule emphasizes the importance of using names that accurately describe the purpose and functionality of variables and functions. Clear and meaningful names enhance code readability, making it easier for other developers to understand the code's intent and maintain it effectively.

2. Rule: Limit the length of lines and keep them readable.
   Explanation: This rule suggests maintaining a maximum line length to enhance code readability. Long lines can be difficult to comprehend, especially when viewed on smaller screens. By keeping lines shorter and concise, code becomes more readable, and it becomes easier to understand the code's structure and logic.

3. Rule: Avoid using global variables whenever possible.
   Explanation: This rule encourages developers to minimize the usage of global variables. Global variables can lead to naming conflicts, make code harder to reason about, and increase the chances of unintended side effects. By encapsulating variables within functions or using modular approaches, code becomes more modular, maintainable, and less prone to bugs.

4. Rule: Always use curly braces for code blocks, even if they contain a single statement.
   Explanation: This rule promotes consistent coding style by mandating the use of curly braces for code blocks, even if they contain only a single statement. This practice helps avoid potential bugs due to missing or misplaced braces and enhances code readability, especially when additional statements are added in the future.

These are just a few examples of codified style guide rules that help establish consistent coding practices and improve code quality in terms of readability, maintainability, and reducing the likelihood of errors.

---

6. To date, what bug has taken you the longest to fix - why did it take so long?
Creating a search button in the final capstone for IWA 19 was by far the longest to fix, it took me a long time because I required a lot of eventListeners and it also contained a submit button, those were my challenges with the bug.
