DWA_01.3 Knowledge Check_DWA1

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

a) Maintainability and Readability: As software systems grow in size and complexity, managing that complexity becomes crucial for maintaining the software over time. The software can become difficult to understand, modify, and debug without proper management, leading to increased maintenance costs and reduced productivity. And if there is a bug in the code or errors it will be easy for developers to see and get it fixed. Managing complexity improves the readability for humans and the comprehensibility of the code. Clean and well-structured code is easier for developers to understand and more manageable, reducing the time required for onboarding new team members, collaborating on projects, and maintaining codebases.
c) Debugging and troubleshooting: Complex software systems can make it challenging to identify and fix bugs. By managing complexity, code becomes more modular and easier to isolate issues, reducing the time and effort required to debug and troubleshoot problems.
d) Scalability and extensibility: A well-managed codebase is more scalable and extensible. It allows for easier integration of new features, modules, or components without disrupting the existing functionality. Proper management of complexity helps future-proof software systems and supports their growth

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

- Size and scope of the codebase: Large codebases with numerous files, functions, and classes can lead to increased complexity.
- Dependencies and interactions: Complex software often arises from intricate dependencies and interactions between different components or modules.
- Poor architecture and design: Inadequate software architecture and design choices can result in unnecessary complexity.
- Lack of documentation and comments: Insufficient documentation and comments make it harder to understand the purpose and behaviour of code.
- Mismatching global variables.
- Accumulated technical debt: Taking shortcuts or implementing quick fixes without proper refactoring can accumulate technical debt and increase complexity.

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

- Modularization: Breaking down code into smaller, self-contained modules help reduce complexity and improve maintainability.
- Abstraction and encapsulation: Using proper abstractions and encapsulation techniques improves code organization and reduces complexity.
- Code organization and naming conventions: Consistent code organization and meaningful naming conventions enhance code readability and comprehension.
- Code reviews and refactoring: Regular code reviews and refactoring sessions help identify and address complex code

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

1. Reduced code readability: Complex code is often difficult to read and understand. On a small scale, this might lead to confusion and errors for the original developer and can pose challenges for future maintenance or collaboration.
2. Increased debugging time: Unmanaged complexity can make it harder to locate and fix bugs. Without clear code structure and logical flow, tracking down issues becomes time-consuming and frustrating.
3. Reduced maintainability: Complexity can hinder the ability to maintain the code effectively. It becomes harder to make changes or add new features without introducing unintended side effects or breaking existing functionality.
4. Reduced productivity: Unmanaged complexity can slow down development processes. Developers spend more time trying to decipher complex code, leading to decreased productivity and slower delivery of new features or bug fixes, and it can cause a company to go bankrupt.
5. Higher risk of errors: Complex code is more prone to errors. When code is not well written and lacks clear structure, it becomes challenging to anticipate and prevent potential issues, resulting in more bugs and software defects.

---

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

Airbnb Style Guide:

The Airbnb JavaScript style guide provides a comprehensive set of guidelines and recommendations for writing JavaScript code. It aims to improve code readability, maintainability, and consistency across projects.

Standard Javascript Style Guide:

The Standard JavaScript styles, also known as "JavaScript Standard Style," follow a simple and minimalistic approach to code formatting and style.

---

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

When I was doing my IWA19 my imports were not working, a struggled for a day or so only to find out my scripts wasn’t properly linked with my HTML file and I didn’t use defer.
