DWA_01.3 Knowledge Check_DWA1

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

- **Scaling:** To ensure that a system can handle growth effectively, scaling code is important because complex software systems often need to be scaled to be able to handle the amount of data.

- **Maintenance:** As the code or software system grows and evolves, complexity can make it hard to read, understand and modify code. By effectively managing the complexity of a code, the developer can ensure that the codebase remains maintainable which will allow for easier debugging and code improvement.

- **TeamWork:** Managing the complexity of code is crucial in effective code collaboration. When a code is not readable or hard to understand it makes it difficult for team members to work together properly, which may lead to project delays and unnecessary errors.

---

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

- **Time**: When a developer is under pressure to deliver results quickly, they can make mistakes by taking shortcuts and not having enough time to refactor and simplify the code.

- **Poor code design**: If software lacks clarity or isn't designed well, it can become complex.

- **Documentation and not commenting:** For easy reading and understanding, it is good practice to have comments and documentation to help the next developer to have a good understanding of the use of code.

---

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

- **Modularization:** Breaking the code into smaller, self-contained modules helps manage code complexity. Using modules allows developers to encapsulate functionality and promote reusability.

- **Documentation and comments:** The use of proper documentation and comments helps manage software by providing explanations and guidelines making it easier to understand.

---

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

- **Reduced Readability:** Complex code can be hard to read and understand. It may also lack the flow of logic.

- **Increased debugging Time**: Because the complexity is not managed, it becomes harder to fix and identify bugs, which can Slow down production.

- **Unable to reuse the code:** The lack of modularity makes it harder to reuse code blacks in different parts of the project or in another project.

- **Decreased Scalability :** When the codebase becomes compact and difficult to change or modify, It can become difficult for the system to handle increased demands.

---

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

- **Use camelCase for variables and function names**.
  In JS, one of the ways to name things is to use camelCasing, where multiple words are joined together and each word starts with a capital letter, every word except the first word.
  E.g. thisIsCamelCasing
  'this' is the first word so it starts with a small letter but every word after that starts with a capital letter

- **Consistency in naming style:**
  Pick a naming style and stick to it

- **Commenting:**
  Use /**..... */ for multiple line and // for single line comments

- **Prefixing your comments with FIXME or TODO** :
  This will help other developers quickly understand if you're pointing out a problem that needs to be revisited, or if you're suggesting a solution to the problem that needs to be implemented. These are different than regular comments because they are actionable. The

actions are FIXME: -- need to figure this out or TODO: -- need to implement.

---

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

- **Spelling error - 2days**
  I wasn't aware of an error that was preventing my CSS file from linking to my HTML file it was the difference between ‘rel’ and ‘ref’
