DWA_01.3 Knowledge Check_DWA1

It contributes to the future success of the project by making things like maintenance, scalability, development and adapting easier over the softwares lifetime.

---

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

There are many factors that create complexity in software. There are interdependencies, algorithmic complexity, legacy code and technical debt, size and scale and concurrency and parallelism to name but a few.

---

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

There are many ways in which to manage complexity in JavaScript that make it more maintainable and improves its scalability over its lifecycle. These techniques include modularization, dependency management, encapsulation , inheritance, polymorphism and good documentation of your work.

---

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

There are several implications of not managing complexity at a small scale that will become evident as the project progresses. This could be the introduction of buggy code, the code becomes less readable and maintainable and it starts to lack scalability. This all increases the production time of the software and makes it more resistant to change further down the line and limiting the softwares viability in the long run.

---

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

The two style guides mentioned in this module are the JavaScript style guide and the Airbnb style guide.

The JavaScript style guide:

Uses camelCase for naming of variables and functions.
Uses uppercase and underscores when naming constants.
For every level of indentation use 2 spaces.
Semicolons are used to end statements.
Use single quotes for strings except in specific cases.
Spaces are used constantly to improve readability.
Prefers use of function declarations over funcion expressions.
Consistent formatting for objects and arrays.
Explain complex pieces of code or give context using comments.
When appropriate use ES6 features.
For importing and exporting use ES6 module syntax.
When handling errors use try-catch blocks.

AirBNB style guide:
Use ‘let’ only if the value needs to be reassigned otherwise use const when declaring your values.
Object shorthand syntax is used when creating objects.
Arrow functions are preferred when creating anonymous function expressions.
‘extend’ is used for inheritance and ‘class’ is used for object constructors.
For multiple value returns and function parameters use destructuring.
Use 2 spaces for indentation and consistently space your operators.
Use single quotes for string except where a string contains a single quote.
When writing comments be sure they are meaningful and use JSdoc for documenting your functions.
Avoid wildcard imports and name your exports as opposed to resorting to default exports.
When using multiline object literals and arrays make sure to include trailing commas.

---

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

I don’t really have a specific bug that jumps out as having taken me the longest, but I usually encounter bugs in my code due to me either not knowing the correct terminologies when writing my code or when I try to implement new code that I don’t really understand.

---

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