DWA_04.3 Knowledge Check_DWA4

---

1. Select three rules from the Airbnb Style Guide that you find **useful** and explain why.

- Prefer const over let and let over var - This rule encourages the use of const for variables that won't be reassigned and let for variables that will be reassigned. It helps improve code readability and maintainability by enforcing immutability whenever possible, reducing the risk of accidental reassignments and unexpected behavior. Because 'var' is a function scope. And 'const' and 'let' are blocked scoped.
- Use template literals instead of string concatenation - Template literals provide a more concise and readable way to concatenate strings, especially when dealing with complex string interpolation. They allow for easy embedding of variables and expressions within the string, making the code more expressive and reducing the chances of introducing syntax errors.
- Use arrow functions when you need lexical this - Arrow functions inherit this value from the surrounding context, eliminating the need for explicit binding or using workarounds like .bind(this). They provide a more concise syntax and help avoid common pitfalls related to the dynamic nature of this, making code easier to reason about.

---

2. Select three rules from the Airbnb Style Guide that you find **confusing** and explain why.

- Using function expressions instead of function declarations -This rule suggests using function expressions over function declarations. While there are valid reasons for this preference, such as avoiding hoisting issues, the distinction between function expressions and declarations can be confusing for beginners.
- Avoid using the arguments object - This rule advises against using the arguments object inside functions and suggests using rest parameters (...args) instead. While rest parameters offer more flexibility and clarity, the arguments

object can still be useful in certain scenarios, especially when dealing with non-arrow functions or dynamically determined number of arguments.

- Do not use bitwise operators - This rule discourages the use of bitwise operators (&, |, ^, etc.) as they can lead to code that is less readable and prone to errors. However, bitwise operators have specific use cases, such as performing low-level bit manipulation or optimizing certain algorithms. Understanding when and how to use them appropriately is important for more advanced programming scenarios.
