DWA_04.3 Knowledge Check_DWA4

---

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

Variable declarations: Use const or let for variable declarations; avoid using var. To avoid redeclaring a variable by mistake.
Use arrow functions instead of function expressions or function declarations for inline functions (prefer-arrow-callback): This rule suggests using arrow functions for inline functions, which can lead to more concise and readable code. Arrow functions have a lexically bound this value, making it easier to access the correct context within the function.

Use === and !== over == and != : This rule encourages the use of strict equality operators (=== and !==) instead of loose equality operators (== and !=) for equality comparisons in JavaScript. The use of strict equality operators is useful because it makes your code more explicit and self-explanatory. When you use === and !==, it clearly indicates that you are comparing both value and type, leaving no room for uncertainty. This improves code readability and makes the intent of the comparison more apparent to other developers who read your code.

Use single quotes for strings unless needing double quotes: This rule promotes consistency in string representation. Using single quotes for strings is generally preferred in JavaScript, unless double quotes are necessary within the string itself. It is useful because of its clarity and differentiation. Differentiating between strings and code can be easier when using single quotes. When you use double quotes for strings, it can be less obvious when a quote character is used for string delimiters or when it's part of the string itself. Single quotes provide a clearer visual indication of the string's boundaries.

---

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

Disallow using the arguments object (no-args): This rule discourages using the arguments object, which is a special JavaScript object that holds all the arguments passed to a function. The confusion may arise because the arguments object can be useful in certain scenarios, particularly for functions that accept a variable number of arguments. Developers who are not aware of the potential issues with using arguments might question why its usage is disallowed.

Avoid using unary increments and decrements (++, --).
This rule suggests avoiding the use of the increment (++) and decrement (--) operators and encourages the use of explicit assignment (+= 1 or -= 1) instead. I find this rule confusing because the increment and decrement operators are common and concise ways to modify variables by one. While there can be readability concerns when used excessively or in complex expressions, wise use of ++ and -- can sometimes improve code readability and conciseness.

Disallow direct modification of Object.prototype (no-extend-native): This rule disallows extending native JavaScript objects like Object.prototype, Array.prototype, or String.prototype. The confusion may arise because modifying these prototypes can have unintended consequences and introduce potential bugs. However, developers who are not aware of this rule may not understand why their attempts to extend these prototypes are disallowed.
