DWA_04.3 Knowledge Check_DWA4

---

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

Rule: "Use const for all of your references; avoid using var."
Explanation: This rule promotes the use of const over var for declaring variables. Using const helps enforce immutability and prevents accidental reassignment of variables. It improves code readability and reduces the risk of introducing bugs due to unintended variable modifications.

Rule: "Always use arrow functions (=>) for function expressions and callbacks."
Explanation: Arrow functions provide a concise syntax and lexical scoping of this, which can help avoid confusion and potential bugs caused by the dynamic scoping of traditional function expressions. They also provide a more functional programming style and enhance code readability by reducing the verbosity of function definitions.

Rule: "Avoid using the Function constructor for dynamic code execution."
Explanation: This rule discourages the use of the Function constructor for dynamically evaluating code. The use of Function constructor can introduce security vulnerabilities and make code harder to understand and maintain. Instead, it encourages the use of other approaches like using closures or higher-order functions for dynamic code execution, which are generally safer and more maintainable.

---

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

Rule: "Only include a single import statement per module."
Explanation: This rule suggests having only one import statement per module rather than grouping multiple imports together. While it can help improve clarity and make dependencies more explicit, there may be cases where it is more practical or logical to

have multiple related imports in a single statement. This rule might be confusing because it doesn't account for different use cases and may require additional refactoring in some scenarios.

Rule: "Do not use wildcard imports."
Explanation: This rule discourages the use of wildcard imports, such as import * as moduleName from 'module'. While it is generally recommended to have explicit imports to enhance code clarity and maintainability, there may be situations where wildcard imports are convenient, especially when dealing with large libraries or utility modules. The rule might be confusing because it is presented as an absolute prohibition without considering potential valid use cases.

Rule: "Avoid using the with statement."
Explanation: This rule advises against using the with statement, which allows for convenient access to object properties without explicit qualification. While with can lead to potential scoping issues and make code harder to understand, there might be rare cases where it can be used judiciously. The rule might be confusing because it doesn't provide clear guidance on when it is acceptable or situations where with could be used safely.
