DWA_04.3 Knowledge Check_DWA4

---

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

- **15.6 Ternaries should not be nested and generally be single line expressions.**

  This rule provides a standard for structuring ternaries, especially when the logic becomes more complex (the ternary expression is long or contains one or more nested ternary expressions).

  I often use ternaries with simple expressions and avoid them when the expression is complex (contains nested ternaries) or is lengthy. However, with this particular style guide rule, I have the confidence and understanding to ensure that my use of this syntax aligns with a certain standard.

- **15.7 Avoid unneeded ternary statements.**

  This rule offers clarity on when to use ternaries. I have often felt uncertain about whether to use a ternary or logical operators in certain situations. After reading this style guide rule, I now have a clearer understanding.

- **13.7 Avoid linebreaks before or after = in an assignment. If your assignment violates max-len, surround the value in parens.**

  I have had the Prettier formatter format some of my code in this manner in the previous challenge. I assumed it was an accepted practice because it was done by the code formatter. However, it didn't make sense to me, as I had never seen such a case anywhere else. I attempted to manually adjust the assignment values to avoid the line break after the assignment operator in a logical manner, but the code formatter still formatted it this way.

---

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

- **18.4 Prefixing your comments with `FIXME` or `TODO` helps 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`.**

  I find this rule confusing because it raises questions about how these comments are actionable and whether there's a specific way or syntax to access them. I haven't encountered this syntax commonly in JavaScript commenting, so it leaves me wondering if it's a standard convention. If not, I'd like to know what alternative methods exist for indicating issues or proposing solutions in JavaScript.

- **23.9 Acronyms and initialisms should always be all uppercased, or all lowercased. Why? Names are for readability, not to appease a computer algorithm.**

  Does this mean developers can disregard the standard naming convention when it comes to acronyms and use them in an unconventional manner?

  How should acronyms be handled when they are the entire variable name or form the last part of the variable name? This scenario can potentially be confusing.

- **11.1 Don't use iterators. Prefer JavaScript's higher-order functions instead of loops like for-in or for-of. Why? This enforces our immutable rule. Dealing with pure functions that return values is easier to reason about than side effects. `Use map() / every() / filter() / find() / findIndex() / reduce() / some() / ...` to iterate over arrays, and `Object.keys() / Object.values() / Object.entries()` to produce arrays so you can iterate over objects.**

  Does this mean developers should completely abstain from using iterators in their code, or are there specific cases where they might still be appropriate?

  Is creating a deep copy of an array before iterating over it a common practice to ensure immutability?

  Are there any scenarios or exceptions where using iterators might be preferred or more suitable?