DWA_04.3 Knowledge Check_DWA4

---

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

23.2 Use camelCase when naming objects, functions, and instances.

- Having a standardised naming rule makes for cleaner code.
- I also prefer camelCase for naming these things

24.3 If the property/method is a `boolean`, use `isVal()` or `hasVal()`.

- Just makes grammatical sense to me

30.1 Yup.

- Just… Yup
- But for real, it's the section on testing and just saying that testing is important and devs should always be writing tests.

---

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

- 22.2 Strings: eslint: `no-new-wrappers`

```
// => this.reviewScore = 9;

// bad
const totalScore = new String(this.reviewScore); // typeof totalScore is
"object" not "string"

// bad
const totalScore = this.reviewScore + ''; // invokes this.reviewScore.valueOf()

// bad
const totalScore = this.reviewScore.toString(); // isn't guaranteed to return a
string

// good
```

- `const totalScore = String(this.reviewScore);`

- Is there a reason I've missed that using `${whateverValue}` would be incorrect?

Semicolons

- 21.1 Yup.

  Why? When JavaScript encounters a line break without a semicolon, it uses a set of rules called Automatic Semicolon Insertion to determine whether it should regard that line break as the end of a statement, and (as the name implies) place a semicolon into your code before the line break if it thinks so. ASI contains a few eccentric behaviors, though, and your code will break if JavaScript misinterprets your line break. These rules will become more complicated as new features become a part of JavaScript. Explicitly terminating your statements and configuring your linter to catch missing semicolons will help prevent you from encountering issues.

- Why do they enforce this and we haven't had to do this?
- It just feels unnecessary and strict.

20.2 Additional trailing comma: Yup. eslint: `comma-dangle`

Why? This leads to cleaner git diffs. Also, transpilers like Babel will remove the additional trailing comma in the transpiled code which means you don't have to worry about the trailing comma problem in legacy browsers.

```
// bad - git diff without trailing comma
const hero = {
     firstName: 'Florence',
-    lastName: 'Nightingale'
+    lastName: 'Nightingale',
+    inventorOf: ['coxcomb chart', 'modern nursing']
};

// good - git diff with trailing comma
const hero = {
     firstName: 'Florence',
     lastName: 'Nightingale',
+    inventorOf: ['coxcomb chart', 'modern nursing'],
};
```

- In my experience with my own errors, the code never ran when I forgot to put a trailing comma anyway, so why does it need it’s own rule?
