DWA_04.3 Knowledge Check_DWA4

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

1. Use consistent indentation (space or a tab).
    a. Indentation makes it easier for me to read my code, plus it makes it looks neat
2. The use template literals for string concatenation.
    a. It is easier to use interpolation for string concatenation plus interpolation does not confuse when I have to use a quoted text in my code

    ```
    // This is a  bad practise
    const greeting = 'How are you, ' + firstName + '?';
    // This is good practise
    const greeting = `How are you, ${firstName}?`;
    ```

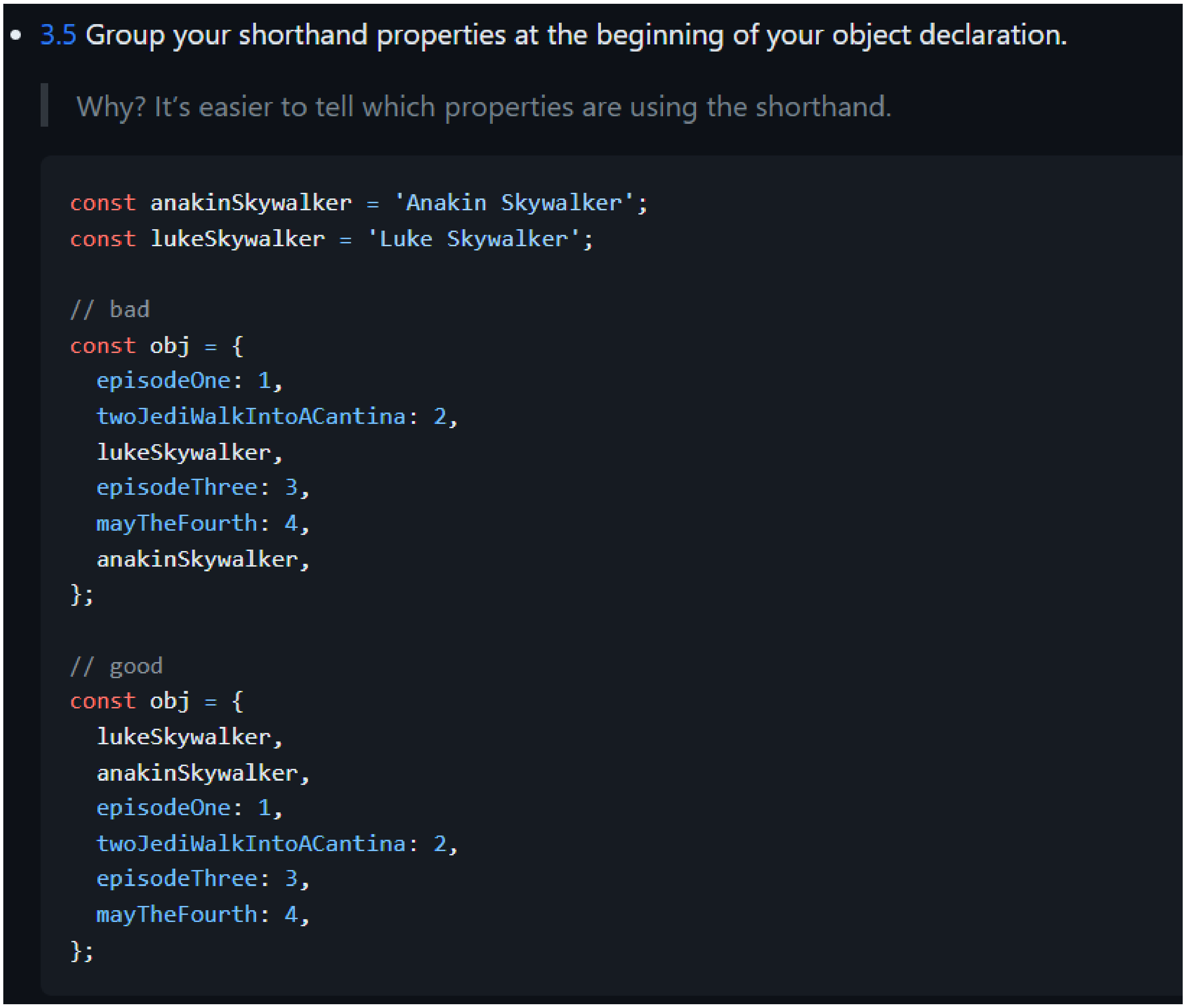
3. Using literal declarations
    a. Literal declarations allow us to declare all properties of an object or an array in one location. Also, it is better to declare a variable without a dot notation. For example:

    ```
    // This is a bad practice
    const person = new Object();
    object.firstName = 'George';

    //This is a good practice
    const person = {
     firstName: 'George',
    };
    ```

---

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

1. 

- 3.5 Group your shorthand properties at the beginning of your object declaration.

> Why? It's easier to tell which properties are using the shorthand.

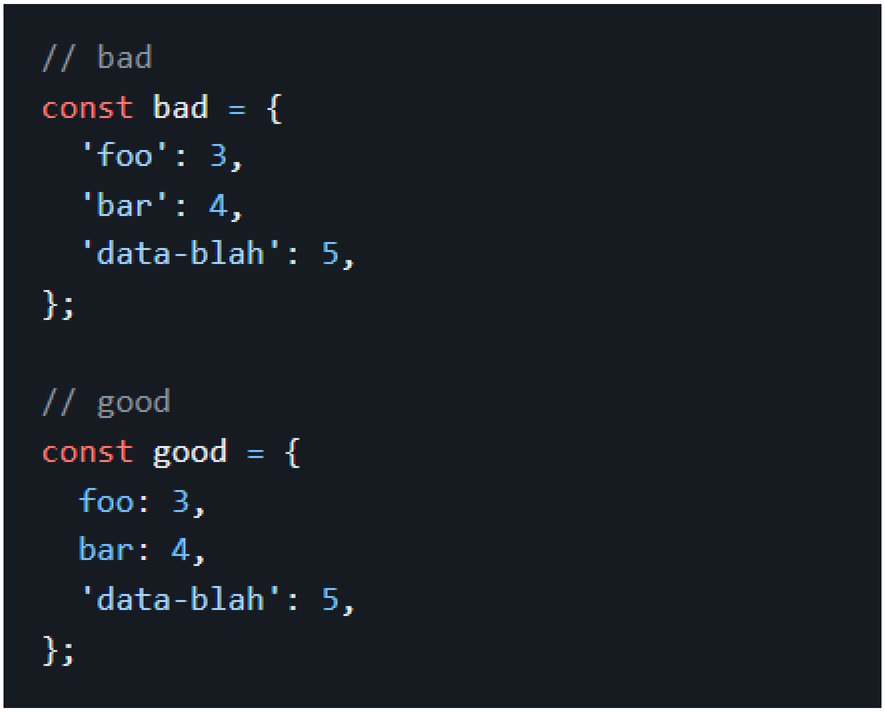
```
const anakinSkywalker = 'Anakin Skywalker';
const lukeSkywalker = 'Luke Skywalker';

// bad
const obj = {
  episodeOne: 1,
  twoJediWalkIntoACantina: 2,
  lukeSkywalker,
  episodeThree: 3,
  mayTheFourth: 4,
  anakinSkywalker,
};

// good
const obj = {
  lukeSkywalker,
  anakinSkywalker,
  episodeOne: 1,
  twoJediWalkIntoACantina: 2,
  episodeThree: 3,
  mayTheFourth: 4,
};
```

Does not doing you code this way affects it's functionality or not?

2. Only quote properties that are invalid identifiers

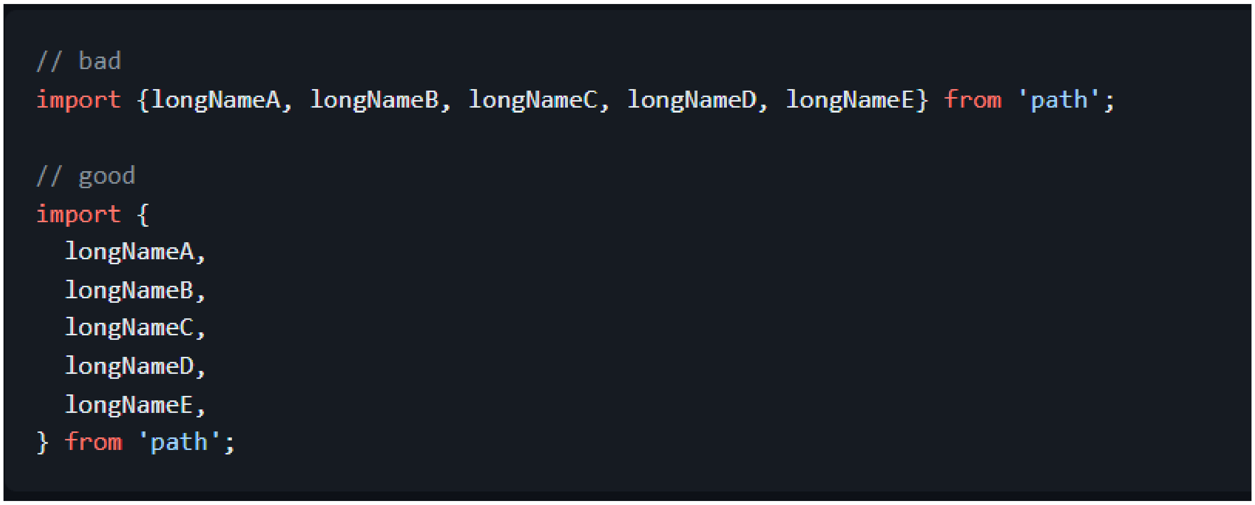


```
// bad
const bad = {
  'foo': 3,
  'bar': 4,
  'data-blah': 5,
};

// good
const good = {
  foo: 3,
  bar: 4,
  'data-blah': 5,
};
```

What is the difference? Will it throw an error if not used?

3. Multiline imports indentation just like a multiline array and object literals.



```
// bad
import {longNameA, longNameB, longNameC, longNameD, longNameE} from 'path';

// good
import {
  longNameA,
  longNameB,
  longNameC,
  longNameD,
  longNameE,
} from 'path';
```

The two seem to work the same and the first one makes your code shorter so why is the second one preferred over the first one?