JavaScript Coding Conventions

Taken from W3Schools, AirBnB

Object Rules

General rules for object definitions:

- Place the opening bracket on the same line as the object name.
- Use colon plus one space between each property and its value.
- Use quotes around string values, not around numeric values.
- Do not add a comma after the last property-value pair.
- Place the closing bracket on a new line, without leading spaces.
- Always end an object definition with a semicolon.

```
var person = {
     firstName: "John",
     lastName: "Doe",
     age: 50,
     eyeColor: "blue"
};
```

Short objects can be written compressed, on one line, using spaces only between properties, like this:

```
var person = {firstName:"John", lastName:"Doe", age:50,
eyeColor:"blue"};
```

Naming Conventions

Always use the **camelCase** naming convention for variable and function names

Ex:

```
firstName = "John";
lastName = "Doe";

componentDidMount() {
}
```

Comments

Use /** ... */ for multi-line comments.

Ex:

```
// bad
// make() returns a new element
// based on the passed in tag name
//
// @param {String} tag
// @return {Element} element
function make(tag) {
  // ...
  return element;
}

// good
/**
 * make() returns a new element
 * based on the passed-in tag name
 *
 * @param {String} tag
 * @return {Element} element
 */
function make(tag) {
  // ...
  return element;
}
```

Use // for single-line comments. Give every comment their own line.

```
//bad
const active = true; //is current tab

//good
//is current tab
const active = true;
```

Documentation

Make sure all your functions and variables are readable so it would be easy to read. Use the commenting style mentioned above, and when including description for your functions don’t hesitate from being descriptive. The description should explain the purpose of the function and how to use it. Also you can comment any subtle nuances in code you feel should be documented. Methods should have param and return type descriptions.

Ex:

```
/**
 * make() returns a new element
 * based on the passed-in tag name
 *
 * @param {String} tag
 * @return {Element} element
 */
function make(tag) {
  // ...
  return element;
}
```