JavaScript Coding Conventions

The following coding conventions are mainly based on https://www.w3schools.com/js/js_conventions.asp

Coding conventions are **style guidelines for programming**. They typically cover:

- Naming and declaration rules for variables and functions.
- Rules for the use of white space, indentation, and comments.
- Programming practices and principles

Coding conventions **secure quality**:

- Improves code readability
- Make code maintenance easier

Coding conventions can be documented rules for teams to follow, or just be your individual coding practice.

Variable Names

We use **camelCase** for identifier names (variables and functions).

All names start with a **letter**.

At the bottom of this page, you will find a wider discussion about naming rules.

```
firstName = "Ruby";
lastName = "Nguyen";

price = 30.90;
tax = 0.20;

fullPrice = price + (price * tax);
```

Spaces Around Operators

Always put spaces around operators (= + - * /), and after commas:

```
const x = y + z;
const values = ["Mango", "Apple", "Orange"];
```

Code Indentation

Always use 4 spaces for indentation of code blocks:

```
const toCelsius = (fahrenheit) => {
    return (5 / 9) * (fahrenheit - 32);
}
```

Statement Rules

General rules for simple statements:

- Always end a simple statement with a semicolon.

```
const values = ["Mango", "Apple", "Orange"];

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

General rules for complex (compound) statements:

- Put the opening bracket at the end of the first line.
- Use one space before the opening bracket.
- Put the closing bracket on a new line, without leading spaces.
- Do not end a complex statement with a semicolon.

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.

```
const 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:

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

Line Length < 80

For readability, avoid lines longer than 80 characters.

If a JavaScript statement does not fit on one line, the best place to break it, is after an operator or a comma.

Naming Conventions

Always use the same naming convention for all your code. For example:

- Variable and function names written as **camelCase**
- Global variables written in **UPPERCASE** (We don't, but it's quite common)
- Constants (like PI) written in **UPPERCASE**

Should you use **hyp-hens**, **camelCase**, or **under_scores** in variable names?

This is a question programmers often discuss. The answer depends on who you ask:

Hyphens in HTML and CSS:

HTML5 attributes can start with data- (data-quantity, data-price).

CSS uses hyphens in property-names (font-size).

Underscores:

Many programmers prefer to use underscores (date_of_birth), especially in SQL databases.

Underscores are often used in PHP documentation.

PascalCase:

PascalCase is often preferred by C programmers.

camelCase:

camelCase is used by JavaScript itself, by jQuery, and other JavaScript libraries.