CSS CODING STANDARD RULE DOCUMENT

INTRODUCTION

Coding standards for CSS aim to ensure that stylesheets are consistent, maintainable, and easily readable. They promote best practices and facilitate collaboration among developers.

Table of Contents

---

1. Formatting

1.1. Indentation and Spacing:

- Use two spaces for indentation.
- Avoid trailing whitespace in CSS files.
- Use soft-tabs with a two-space indent.

Example:

```
.selector {
  property: value;
}
```

1.2. Quotes

- Use double quotes for attribute values.

Example:

```
[data-vegetable="liquid"] {
  background-color: goldenrod;
  background-image: url("../../media/examples/lizard.png");
}
```

1.3. Shorthand Notation

- Use shorthand notation where possible.

Example:

```
.example {
  padding: 10px 20px;
}
```

1.4. Spaces

- Put spaces after colons in property declarations.
- Put spaces before opening curly braces in rule declarations.
- Use spaces around the `:`, `{`, and `}` characters.

Example:

```
.selector {
  property: value;
}
```

1.5. Hex Color Codes

- Use hex color codes (`#000`) unless using `rgba()`.

Example:

```
.selector {
  color: #fff;
  background-color: #000; /* Fallback value */
}
```

1.6. Fallback Properties:

- Always provide fallback properties for older browsers.

Example:

```
.selector {
  background-color: #000; /* Fallback value */
  background-image: linear-gradient(black, grey);
}
```

1.7. URLs and Imports:

- Always quote `url()` and `@import()` contents.

Example:

```
.selector {
  background-image: url("image.jpg");
}
```

2. Naming

2.1. Lowercase with Hyphens

- Use lowercase letters with hyphens for IDs, classes, and attributes.

Example:

```
.dataset-list {}
```

3. Comments

3.1. Use CSS-Style Comments

- Use CSS-style comments to explain code that isn't self-documenting.
- Leave a space between the asterisks and the comment.

Example:

```
.selector {
  /* Comment explaining the property */
  property: value;
}
```

4. Modularity and Specificity

4.1. Module-Based Styling:

- Group selectors into modules where possible.
- Avoid having too many selectors in one declaration to enhance readability.

Example:

```
.module {}
.module-item {}
.module-item .element {}
```

4.2. Class-Based Styling

- Use classes for styling to make styles more robust and adaptable to changing HTML.

Example:

```
html
<ul class="social">
  <li><a href="">Twitter</a></li>
  <li><a href="">Facebook</a></li>
  <li><a href="">LinkedIn</a></li>
</ul>


css
.social .twitter {}
.social .facebook {}
.social .linked-in {}
```

4.3. Avoid Tag Names and IDs

- Do not use tag names or IDs in selectors, as they limit re usability and make overriding styles difficult.

Example:

```
.dataset-list {}
```