JavaScript Coding Standards

General Conventions
- **File Naming:** Use camelCase for file names. Example: `myScript.js`
- **Indentation:** Use 2 spaces for indentation.
- **Line Length:** Limit lines to 80 characters.
- **Semicolons:** Always use semicolons.
- **Quotes:** Use single quotes for strings (`'example'`).

Braces and Blocks
- **Function and Conditional Braces:**
  ```javascript
  function myFunction() {
    // code
  }

  if (condition) {
    // code
  } else {
    // code
  }
  ```

Naming Conventions
- **Variables:** Use camelCase for variables. Example: `let myVariable = 1;`
- **Constants:** Use `UPPER_CASE` for constants. Example: `const MAX_SIZE = 100;`
- **Functions:** Use camelCase for functions. Example: `function myFunction() {}`
- **Classes:** Use PascalCase for class names. Example: `class MyClass {}`

Documentation
- **Comments:** Use `//` for single-line comments and `/* */` for multi-line comments.
- **Function Documentation:**
  ```javascript
  /**
   * Brief description of the function.
   *
   * @param {Type} paramName - Description of the parameter.
   * @return {Type} Description of the return value.
   */
  function myFunction(paramName) {
    // code
  }
  ```

Best Practices
- **Strict Mode:** Use `'use strict';` at the beginning of your JavaScript files.
- **Avoid Global Variables:** Minimize the use of global variables.
- **Consistent Naming:** Stick to the naming conventions throughout the codebase.
- **Avoid Deep Nesting:** Refactor code to avoid deep nesting of callbacks or loops.

HTML Coding Standards

General Conventions
- **File Naming:** Use lowercase letters and hyphens for file names. Example: `index.html`
- **Indentation:** Use 2 spaces for indentation.
- **Line Length:** Limit lines to 80 characters.
- **Quotes:** Use double quotes for attribute values (`<input type="text">`).

Structure and Semantics
- **Doctype:** Always declare the doctype.
  ```html
  <!DOCTYPE html>
  ```
- **HTML5:** Use HTML5 semantic elements (`<header>`, `<footer>`, `<article>`, etc.).
- **Indentation:** Indent nested elements.
  ```html
  <div>
    <p>Example</p>
  </div>
  ```

Naming Conventions
- **Classes and IDs:** Use lowercase and hyphens. Example: `class="my-class"`
- **Attributes:** Use lowercase letters. Example: `type="text"`

Documentation
- **Comments:** Use `<!-- -->` for comments.
  ```html
  <!-- This is a comment -->
  ```

Best Practices
- **Accessibility:** Ensure your HTML is accessible. Use `alt` attributes for images and `aria` attributes where necessary.
- **Validation:** Validate your HTML code using the W3C validator.

JSON Coding Standards

General Conventions
- **File Naming:** Use lowercase letters and hyphens for file names. Example: `data.json`
- **Indentation:** Use 2 spaces for indentation.
- **Line Length:** Limit lines to 80 characters.
- **Quotes:** Use double quotes for keys and string values.

Structure
- **Consistent Formatting:** Maintain consistent formatting throughout the file.
  ```json
  {
    "name": "example",
    "version": "1.0.0",
    "dependencies": {
      "package": "^1.0.0"
    }
  }
  ```

Naming Conventions
- **Keys:** Use camelCase for keys. Example: `"myKey": "value"`

Documentation
- **Comments:** JSON does not support comments. Use descriptive keys and values to make the JSON self-explanatory.

Best Practices
- **Validation:** Validate JSON using a JSON validator.
- **Minimize Data:** Keep JSON data as minimal as possible to reduce file size and parsing time.

Adopting a Published Standard
Consider adopting an existing coding standard, such as the Airbnb JavaScript Style Guide, which is widely used and respected in the industry. For HTML, the W3C HTML guidelines can be a good reference, and for JSON, ensure it is always properly formatted and validated.