**Руководство по стилю кода**

**Общие правила**

* **Именование**: Используйте осмысленные и понятные имена для переменных, функций и классов. Примеры хороших имён: calculateTotal, usersList. Избегайте однобуквенных или трудно понимаемых имён.
* **Консистентность**: Соблюдайте единообразие в стиле кода по всему проекту.
* **Комментарии**: Оставляйте комментарии, где необходимо. Комментарии должны объяснять причину написания определённого кода, а не просто описывать, что он делает.
* **Документация**: Документируйте функции и методы для лучшего понимания другими разработчиками. Указывайте входные параметры, возвращаемое значение и основное назначение функции.

**Форматирование кода**

* **Отступы**: Используйте отступы с пробелами или табуляцией, но будьте консистентны. Стандарт - 4 пробела.
* **Длина строки**: Старайтесь не превышать длину строки в 80-100 символов.
* **Пробелы**: Используйте пробелы для улучшения читаемости, например, до и после операторов =, +, - и т.д. ! Это не касается верстки. Атрибуты пишутся без проблеов. Пример: class=”” data-atr=”” и т.д.
* **Кавычки**: В HTML/CSS используйте двойные кавычки "", в JavaScript/TypeScript - одинарные кавычки ''. (Если это не касается переменных в строке)
* **Малые блоки**: Начинайте верстку с малых блоков, которые будут повторяться на сайте, чтобы можно было их переиспользовать. Примеры таких блоков: .button, , .logo, .card**. Также важно вдыелять по ним html, css и js в отдельные файлы. (Аля коллекция)**
* **Разделение блоков**: Разделяйте блоки и технические классы. Примеры блоков: .button, .logo, .card. Примеры технических классов: .\_img, .\_title, .\_text, где описывается поведение повторяющихся элементов и их свойства.
* **Семантическая верстка. Не забываем, что section – это блок информации, который нельзя выдернуть из контекста. Article – блок информации самостоятельный: пост, анонс поста, рекламка и т.д.**
* **(Экспрементально) использовать header и footer в article.**

**JavaScript/TypeScript**

* **Стрелочные функции**: Используйте стрелочные функции вместо обычных функций, когда это возможно.
* **Константы**: Используйте const для переменных, которые не будут изменяться, и let для переменных, которые могут измениться.
* **Промисы и async/await**: Используйте async/await вместо promises для работы с асинхронным кодом.

**CSS/Sass**

* **Классы**: Используйте методологию BEM для наименования классов. Пример: .block\_\_element--modifier.
* **Переменные**: Используйте переменные для цветов, шрифтов и других часто используемых значений.
* **Семантика**: Применяйте семантические HTML-теги и структуры.

**Git**

* **Коммиты**: Делайте маленькие и осмысленные коммиты с понятными сообщениями. Пример: git commit -m "Добавил функцию фильтрации списка пользователей".
* **Ветки**: Создавайте ветки для каждой новой функциональности или исправления ошибок. Пример: feature/new-feature.

**Мини шпаргалка по верстке.**

**header и footer**

<header> — это не только привычная шапка сайта с логотипом и меню, но и «шапка» какой-нибудь статьи или раздела сайта, в которой могут содержаться заголовки, оглавление и так далее.

<footer> — это подвал сайта: с копирайтами, контактной информацией и так далее. Но может использоваться и в других разделах сайта, например в статье, чтобы разместить дополнительную информацию: данные об авторе, дополнительные ссылки и так далее.

**main**

Тег <main> обозначает основное содержание сайта и по спецификации может использоваться на странице только один раз.

**article и section**

Где и когда использовать разные контейнеры:

* <div> — контейнер общего назначения, не обязательно смысловой. Дивы используются для разметки мелких блоков, создания сетки и декоративных эффектов.
* <section> — более крупный логический контейнер, объединяющий содержание по смыслу. Например, блок «О компании», список товаров, раздел личной информации в профиле и так далее.
* <article> — самостоятельный, цельный и независимый раздел документа. Этот раздел можно в неизменном виде использовать в различных местах, в том числе и на других сайтах. Примеры: статья, пост в блоге, сообщение на форуме и так далее.

**nav**

Тег <nav> — специальный элемент для навигационных блоков. Меню и блоки ссылок лучше оборачивать не в дивы, а в навы.

**aside**

<aside> — это дополнительное содержание, не связанное напрямую с основным. Ещё такие блоки часто называют «сайдбарами» или боковыми панелями.

**time**

<time> — специальный тег для обозначения даты и времени. У этого тега есть атрибут datetime — для указания даты в машиночитаемом формате ISO 8601. Содержимое datetime не отображается.

<time datetime="2014-04-20">Вчера</time> мы готовили тренажёр к публикации.

**figure и figcaption**

<figure> — цельный и независимый блок содержания. Внутри этого тега размещают демонстрационный материал: изображения, схемы, куски кода и так далее.

**Мини шпаргалка по структуре.**

* если нет секций полноформатных:

<head></head>

<body>

<div class="wrapper">

<header></header>

<main>

<div class="container">

<section></section>

<section></section>

<section></section>

</div>

</main>

<footer></footer>

</div>

</body>

* Если есть широкоформатные секции то:

<head></head>

<body>

<div class="wrapper">

<header></header>

<main>

<section>

<div class="container"></div>

</section>

<section>

<div class="container"></div>

</section>

<section></section> // Широкоформатная секция

</main>

<footer></footer>

</div>

</body>