Правила оформления CSS-кода

1. Синтаксис

1.1 В конце строки должна стоять точка с запятой

После пары *свойство: значение* обязательно ставится точка с запятой. Без этого знака препинания не будет работать правило в этой строке и следующее за ним.

Хорошо: после каждого значения стоит точка с запятой

CSS

```css
.selector {
  color: red;
  background-color: gray;
}
```

Плохо: после первого свойства пропущена точка с запятой

CSS

```css
.selector {
  color: red
  background-color: gray;
}
```

1.2 Для отступов внутри правил используйте два пробела

Правила, которые перечисляются внутри фигурных скобок, должны отстоять от начала строки. Для этого используйте 2 пробела. Это позволяет сразу видеть блоки свойств, относящихся к одному селектору.

Хорошо

CSS

```css
.selector {
  color: red;
  background-color: gray;
}
```

Плохо

CSS

```css
.selector {
color: red
   background-color: gray;
}
```

1.3 Значение цветов не сокращается

Если цвет задан в шестнадцатиричной системе, то значение не сокращается, а пишется полностью код из всех шести символов. Для записи используйте строчные буквы. Например, `#e3e3e3`.

Хорошо

CSS

```css
.selector {
  box-shadow: 0 1px 2px #cccccc, inset 0 1px 0 #ffffff;
}
```

Плохо

CSS

```css
.selector {
  box-shadow: 0 1px 2px #CCC, inset 0 1px 0 #FFF;
}
```

1.4 Все пишется строчными буквами

Все названия тегов и свойства пишутся строчными буквами.

Хорошо

CSS

```css
section {
  padding: 15px;
  margin-bottom: 10px;
}
```

Плохо

CSS

```css
sEctiOn {
  PADDING: 15px;
  Margin-Bottom: 10px;
}
```

1.5 Нули не пропускаются

Если число дробное и начинается с нуля, то он не опускается (например, `.5` вместо `0.5`).

Хорошо

CSS

```css
.selector {
  background-color: rgba(0, 0, 0, 0.5);
  opacity: 0.7;
}
```

Плохо

CSS

```css
.selector {
  background-color: rgba(0, 0, 0, .5);
  opacity: .7;
}
```

1.6 Используйте двойные кавычки

В стилях всегда без исключения используются двойные кавычки. Если кавычки не обязательны, то они пишутся все равно.

Хорошо

CSS

```css
.selector[type="text"] { ... }
```

Плохо

CSS

```css
.selector[type=text] { ... }
```

1.7 Пробел после двоеточия

В правилах после двоеточия ставится один пробел (`top: 10px;`). При этом перед двоеточием пробел не нужен.

Хорошо

CSS

```css
.selector {
  padding: 15px;
  margin-bottom: 10px;
}
```

Плохо

CSS

```css
.selector {
  padding:15px;
  margin-bottom: 10px ;
}
```

1.8 Пробелы после запятой в цветах

После запятых внутри значений `rgb()`, `rgba()`, `hsl()`, `hsla()` или `rect()` пробелы ставятся. Это улучшает читаемость.

Хорошо

CSS

```css
.selector {
  background-color: rgba(0, 0, 0, 0.5);
}
```

Плохо

CSS

```css
.selector {
  background-color: rgba(0,0,0,0.5);
}
```

1.9 Пробел до и после комбинатора

Между селекторами до и после комбинатора (например, `p > a`) ставится один пробел.

Хорошо

CSS

```css
ol > li { ... }
```

Плохо

CSS

```css
ol>li { ... }
```

1.10 Каждое свойство с новой строки

Одно свойство — одна строка. Каждое объявление в правиле пишется на новой строке.

Хорошо

CSS

```css
.selector {
  color: red;
  background-color: gray;
  padding: 15px;
  margin-bottom: 10px;
}
```

Плохо

CSS

```css
.selector {
  color: red; background-color: gray; padding: 15px; margin-bottom: 10px;
}
```

1.11 Пробел перед фигурной скобкой

После селектора и перед открывающейся фигурной скобкой ставится один пробел. После открывшейся скобки запись идёт с новой строки.

Хорошо

CSS

```css
.selector {
  color: red;
  background-color: gray;
  padding: 15px;
  margin-bottom: 10px;
}
```

Плохо

CSS

```css
.selector{color: red;
  background-color: gray;
  padding: 15px;
  margin-bottom: 10px;
}
```

1.12 Закрывающая фигурная скобка на новой строке

Закрывающая фигурная скобка после набора свойств пишется на новой строке и без отступа. Она должна быть на одном уровне с селектором. Следующее после этого правило отделяется пустой строкой.

Хорошо

CSS

```css
.selector {
  color: red;
}

.selector-item {
  color: black;
}
```

Плохо

CSS

```css
.selector {
  color: red;}
.selector-item {
  color: black;
  }
```

1.13 Опускайте единицы измерения

Единицы измерения не нужно писать там, где без них можно обойтись. Например, `border: 0`.

Хорошо

CSS

```css
.selector {
  border: 0;
  box-shadow: 0 1px 2px #cccccc, inset 0 1px 0 #ffffff;
}
```

Плохо

CSS

```css
.selector {
  border: 0px;
  box-shadow: 0px 1px 2px #cccccc, inset 0px 1px 0px #ffffff;
}
```

2. Порядок свойств

Порядок логически связанных свойств должен быть сгруппирован следующим образом:

1. Позиционирование
2. Блочная модель
3. Типографика
4. Оформление
5. Анимация
6. Разное

Позиционирование следует первым, поскольку оно влияет на положение блоков в потоке документа. Блочная модель определяет размеры и расположение блоков и идёт следующей.

Все прочие объявления, которые изменяют вид внутренних частей блоков и не влияют на другие блоки, идут в последнюю очередь.

Сгруппированные свойства в правиле отделяются друг от друга пустой строкой.

Порядок объявления подробных правил, таких как `font-size`, `font-family`, `line-height`, должен соответствовать порядку в сокращённой версии правила. В случае совместного использования подробных и сокращённых правил, первой должна идти сокращённая версия.

CSS

```css
.selector-item {
  /* Позиционирование */
  position: fixed;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  z-index: 100;

  /* Блочная модель */
  display: inline-block;
  float: left;
  width: 150px;
  height: 150px;
  margin: 25px;
  padding: 25px;

  /* Типографика */
  font: normal 13px/1.5 "Helvetica", sans-serif;
  font-style: normal;
  font-size: 13px;
  line-height: 1.5;
  font-family: "Helvetica", sans-serif;
  text-align: start;

  /* Оформление */
  color: #999999;
  background-color: #e3e3e3;
  border: 1px solid #333333;
  border-radius: 5px;
  opacity: 1;

  /* Анимация */
  transition: all 0.8s;

  /* Разное */
  will-change: auto;
}
```

3. Имена классов

- Имена классов пишутся строчными буквами, между несколькими словами используется дефис (но не знак нижнего подчёркивания или camelCase). Дефисы служат разделителями во взаимосвязанных классах (например, `.button` и `.button-cancel`).
- Имена должны быть такими, чтобы по ним можно было быстро определить, какому элементу на странице задан класс: избегайте сокращений (единственное исключение — `.btn` для кнопок), но не делайте их слишком длинными (не более трёх слов).
- Для именования классов используются английские слова и термины. Не используйте транслит для названия классов и атрибутов.

Хорошо

CSS

```css
.alert-info { … }
.tweet .user-picture { … }
.button { … }
.layout-center { … }
```

Плохо

CSS

```css
.testElement { … }
.t { … }
.big_green_button { … }
.knopka { … }
```

4. Правило `@import`

Правило `@import` работает медленнее, чем тег `<link>`. В стилях `@import` использовать не желательно.

Хорошо: подключение тегом link

```html
<link rel="stylesheet" href="module.css">
```

Плохо

```html
<style>
  @import url("module.css");
</style>
```

5. Варианты шрифта

Альтернативные варианты шрифта и тип семейства указываются в конце перечисления значений font-family.

В случае использования нестандартных шрифтов обязательно указывать альтернативный веб-безопасный шрифт и тип семейства, чтобы в случае отсутствия нестандартного шрифта в системе изменения внешнего вида страницы были минимальны. Нестандартный и альтернативный шрифты должен быть одного типа.

Порядок шрифтов следующий:

1. нестандартный шрифт;
2. веб-безопасный;
3. тип семейства шрифта.

Список веб-безопасных шрифтов можно посмотреть здесь — cssfontstack.com.

Хорошо: указан альтернативный веб-безопасный шрифт и его тип семейства

CSS

```css
body {
  font-family: "Helvetica", "Arial", sans-serif;
}

/* Кому-то нравится Arial, кому-то Tahoma */
body {
  font-family: "Helvetica", "Tahoma", sans-serif;
}
```

Плохо: указан только нестандартный шрифт

CSS

```css
body {
  font-family: "Helvetica";
}
```

Плохо: указан только нестандартный шрифт и тип семейства, альтернативный веб-безопасный шрифт отсутствует

CSS

```css
body {
  font-family: "Helvetica", sans-serif;
}
```

Плохо: Georgia — шрифт с засечками, а нестандартный шрифт — без засечек

CSS

```css
body {
  font-family: "Helvetica", "Georgia", sans-serif;
}
```