


HTML и CSS

Контейнеры, семантика,
задания стилей



The image features decorative geometric patterns in the top-left and bottom-right corners. These patterns consist of overlapping blue and gold shapes, including rectangles and triangles, some of which are filled with a dotted pattern. The central area is white and contains the main title text.

Блочный и строчный контейнеры

Block vs Inline

<h1>

<p>

<a>

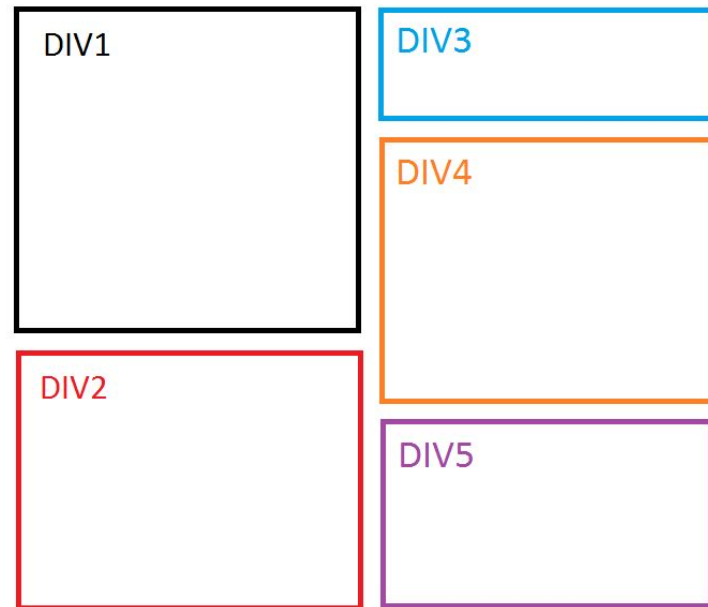
Блочные элементы - это элементы, которые занимают всю строку горизонтального пространства на веб-странице. Это означает, что они автоматически начинаются с новой строки, а последующие элементы автоматически переносятся на новую строку.

Строчные элементы – это элементы, которые являются частью строки и занимают такое количество пространства, которое необходимо для отображения их содержимого.

div - блочный контейнер

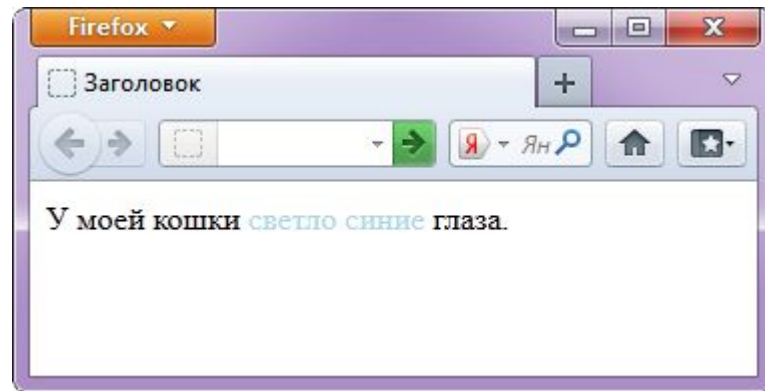
Является универсальным контейнером для контента веб-страницы.

```
<div>Контент</div>
```



span - строчный контейнер

Тег `` предназначен для определения строчных элементов документа. С помощью тега `` можно выделить часть информации внутри других тегов и установить для нее свой стиль.

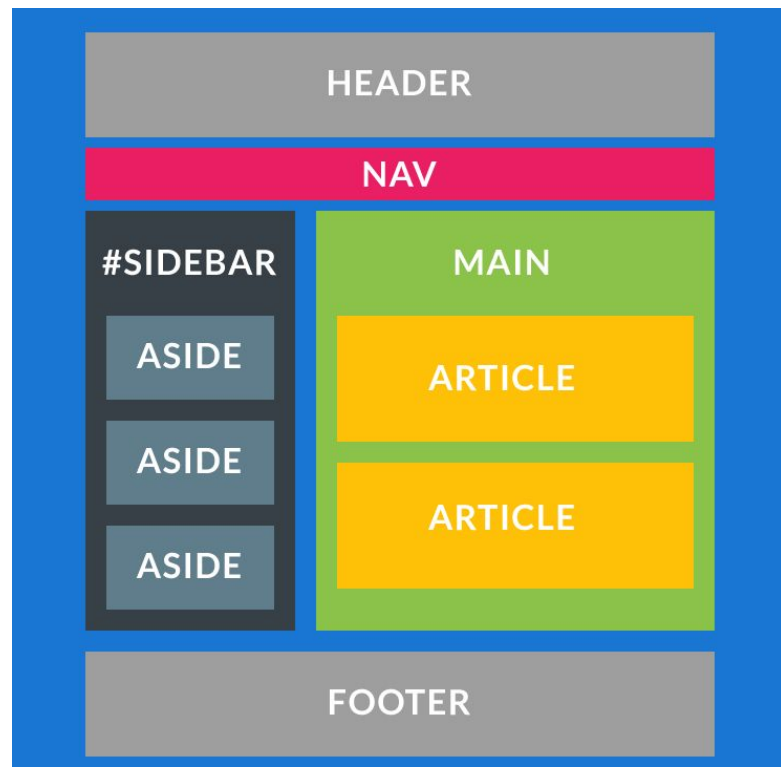


``Контент``

Семантические теги



Семантическая вёрстка —
подход к разметке, который
опирается не на содержание
сайта, а на смысловое
предназначение каждого блока
и логическую структуру
документа.



Для чего нужна семантика:



1. Для accessibility*. Основной инструмент незрячих или частично незрячих людей для просмотра сайтов не браузер, а скринридер, который читает текст со страницы вслух. Этот инструмент «зачитывает» содержимое страницы, и семантическая структура помогает ему лучше определять, какой сейчас блок, а пользователю понимать, о чём идёт речь. Таким образом семантическая разметка помогает большему количеству пользователей работать с вашим сайтом.
2. Чтобы сайт был выше в поисковиках. Поисковики не разглашают правила ранжирования, но известно, что наличие семантической разметки страниц помогает поисковым ботам лучше понимать, что находится на странице, и в зависимости от этого ранжировать сайты в поисковой выдаче.
3. Семантика прописана в стандартах. При работе, например, над поддержкой существующего приложения вам будет проще работать с кодом, написанным по общим стандартам

**Accessibility – это область знания, которая занимается изучением вопросов доступности сайтов, мобильных приложений и программного обеспечения для людей с ограниченными возможностями.*

В html, чтобы выполнить требования семантики используют специальные теги

Ter	Описание
<code><article></code>	Определяет статью
<code><aside></code>	Определяет содержание в стороне от содержимого страницы
<code><details></code>	Определяет дополнительные сведения, которые пользователь может просматривать или скрывать
<code><figcaption></code>	Определяет заголовок для элемента <code><Figure></code>
<code><figure></code>	Задаёт автономное содержимое, например иллюстрации, диаграммы, фотографии, списки кодов и т.д.
<code><footer></code>	Определяет нижний колонтитул для документа или раздела
<code><header></code>	Задаёт заголовок для документа или раздела
<code><main></code>	Указывает основное содержимое документа
<code><mark></code>	Определяет выделенный/выделенный текст
<code><nav></code>	Определяет навигационные ссылки
<code><section></code>	Определяет раздел в документе
<code><summary></code>	Определяет видимый заголовок для элемента <code><Details></code>
<code><time></code>	Определяет дату и время



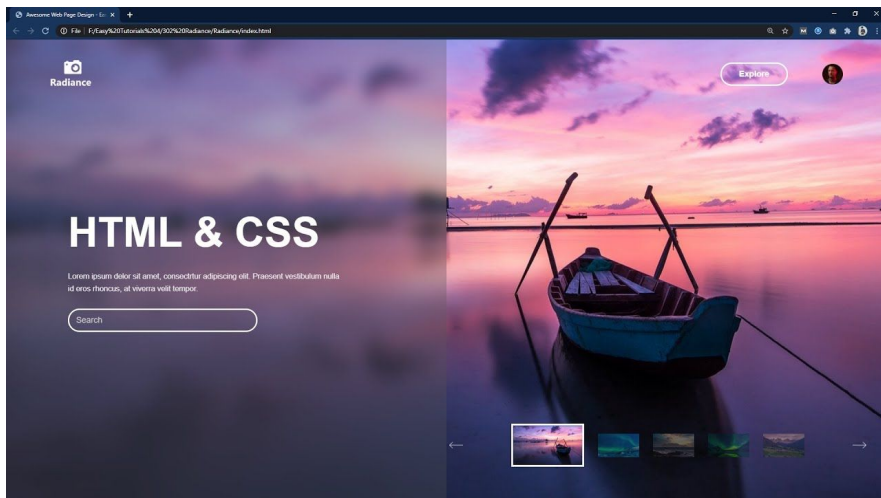


CSS. Введение

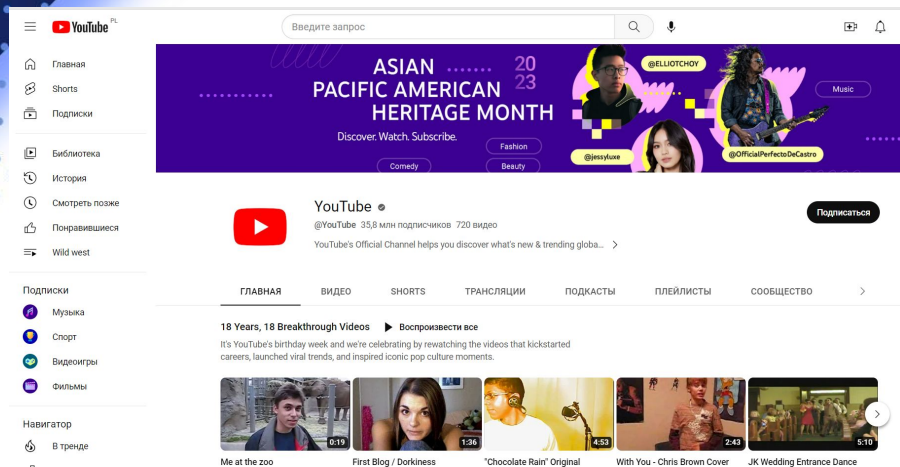
CSS (Cascading Style Sheets) - язык, который отвечает за описание внешнего вида HTML-документа.

Что делает CSS?

Если HTML структурирует контент на странице, то CSS позволяет отформатировать его, сделать более привлекательным для читателя.



HTML + CSS



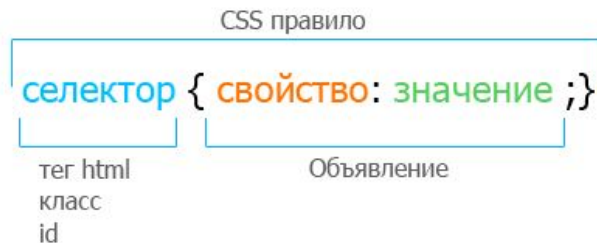
HTML



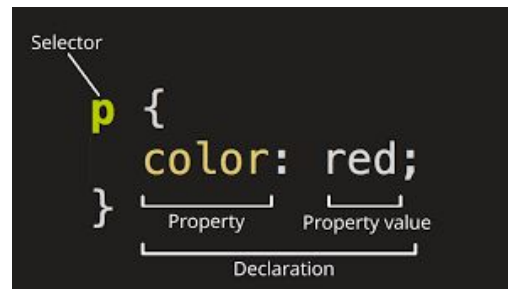
Синтаксис CSS

У языка CSS относительно простой синтаксис. Сначала прописывается селектор — он выбирает конкретный элемент на странице. Потом, после фигурных скобок, указываются свойства со значениями — между ними ставится двоеточие. Сами свойства отделяются друг от друга точкой с запятой.

Схема



Пример



Базовая структура CSS

Документа

```
body {  
  font-family: Arial, Verdana, sans-serif; /* Семейство шрифтов */  
  font-size: 11pt; /* Размер основного шрифта в пунктах */  
  background-color: #f0f0f0; /* Цвет фона веб-страницы */  
  color: #333; /* Цвет основного текста */  
}  
h1 {  
  color: #a52a2a; /* Цвет заголовка */  
  font-size: 24pt; /* Размер шрифта в пунктах */  
  font-family: Georgia, Times, serif; /* Семейство шрифтов */  
  font-weight: normal; /* Нормальное начертание текста */  
}  
p {  
  text-align: justify; /* Выравнивание по ширине */  
  margin-left: 60px; /* Отступ слева в пикселах */  
  margin-right: 10px; /* Отступ справа в пикселах */  
  border-left: 1px solid #999; /* Параметры линии слева */  
  border-bottom: 1px solid #999; /* Параметры линии снизу */  
  padding-left: 10px; /* Отступ от линии слева до текста */  
  padding-bottom: 10px; /* Отступ от линии снизу до текста */  
}
```

CSS Селектор



Что такое селекторы?

Селекторы - это выражения, которые говорят браузеру, к какому элементу HTML нужно применить те или иные свойства CSS, определённые внутри блока объявления СТИЛЯ.

```
h1 {  
  color: blue;  
  background-color: yellow;  
}  
  
p {  
  color: red;  
}
```


Типы базовых CSS селекторов

- Универсальный селектор
- Селекторы по типу элемента
- Селекторы по идентификатору
- Селекторы по классу
- Селекторы по атрибуту



Универсальный селектор

Он применяет стили ко всем элементам страницы и обозначается символом * (звёздочка).

```
* {  
  
    margin: 0;  
  
    box-sizing: border-box;  
  
}
```

Селектор по элементу (тегу)

Этот селектор CSS применяет стили ко всем элементам с одинаковым тегом. Например, для всех `<div>`, `<h2>`, `<p>` и так далее.

```
p {  
    color: red;  
}
```

Селекторы по идентификатору

Селектор по идентификатору обозначается символом **#** (решётка) и применяет стили к элементу, для которого задан атрибут **id** с соответствующим значением. При этом у элемента может быть только один **id**, и этот **id** должен быть уникальным в пределах веб-страницы

html code

```
<p id="article_content">В этой  
статье:</p>
```

css code

```
#article_content {  
  
    font-family: sans-serif;  
  
    font-weight: bold;  
  
}
```

Селекторы по классу

CSS-селектор по классу выбирает элементы, для которых назначен атрибут **class** с соответствующим значением. Селектор класса обозначается символом **.** (точка), после него указывается необходимый класс элемента

html code

```
<p class="base_content">В этой  
статье:</p>
```

css code

```
.base_content {  
    color: blue;  
}
```

Селекторы по атрибуту

Селекторы по атрибуту позволяют выбрать элемент по имени атрибута, его значению или части значения. Селектор атрибута обозначается в квадратных скобках `[]`

1. Селектор по названию атрибута **[attr]**

```
[title] {  
  
    font-weight: bold;  
  
}
```

Селекторы по атрибуту

2. Селектор по имени и значению атрибута **[attr=value]**

```
[title="text"] {  
    font-weight: normal;  
}
```

Селекторы по атрибуту

3. Селектор по началу значения атрибута **[attr^=value]**. Находит элементы с заданным атрибутом, значение которого начинается с value:

```
[href^="link"] {  
  
    font-weight: normal;  
  
}
```


Селекторы по атрибуту

4. Селектор **[attr\$=value]**. Применяет CSS-стили к элементам, у которых значение заданного атрибута оканчивается на value:

```
[href$=".jpg"] {  
  
    font-weight: normal;  
  
}
```

Селекторы по атрибуту

5. Селектор **[attr*=value]**. Селектор по названию атрибута и значению, которое должно содержать value:

```
[title*="content"] {  
    font-weight: normal;  
}
```

Стилевой класс



Создание класса

Для выделения какой-то группы объектов (элементов), которые необходимо наделить одними и теми же свойствами css, необходимо создать класс.

```
.class_name {  
  
    свойство1: значение;  
  
    свойство2: значение;  
  
}
```

Примечание: Имена классов должны начинаться с латинского символа и могут содержать в себе символ дефиса (-) и подчеркивания (_). Использование русских букв в именах классов недопустимо.

Добавление класса

Чтобы добавить класс, необходимо для элементов (тегов), к которым будут применены свойства класса, прописать атрибут `class` с придуманным названием класса в качестве значения атрибута:

```
<h1 class="class_name">Текст сообщения</h1>
```

Возможности классов

- К одному тегу можно добавлять несколько классов, для это нужно перечислить названия в значении атрибута `class` через пробел

```
<h1 class="name1 name2">Текст сообщения</h1>
```

- В селекторе можно указать стили для определённого тега, входящего в класс

```
p.class_name {  
    свойство1: значение;  
    свойство2: значение;  
}
```

Комбинированные стили



Группировка селекторов

Если нескольким селекторам нужно задать одно и то же правило, то можно написать длинно:

```
h1 {  
    color: red;  
}  
  
h2 {  
    color: red;  
}  
  
h3 {  
    color: red;  
}
```

А можно перечислить все селекторы через запятую и написать всего одно CSS-правило:

```
h1, h2, h3 {  
    color: red;  
}
```


Группировка селекторов

При группировке любые виды селекторов записываются в произвольном порядке через запятую и в таком случае правила будут применяться к каждой из групп:

```
h1, .block, #id, h3 {  
    color: red;  
}
```

Объединение

Этот приём применим только для классов и атрибутов, потому что только их может быть больше одного. Селекторы записываются слитно. Стили будут применены только к тому элементу, который содержит все перечисленные селекторы.

```
.class_1.class_2 {  
    color: red;  
}
```

Сложение стилей



Если разные правила для одного элемента содержат свойства, которые не конфликтуют, то они объединяются в один стиль, т.е. каждое новое правило добавляет новую информацию о стиле к тому правилу, которое находится выше:

```
h1 {  
  
    color: gray;  
  
    font-family: sans-serif;  
  
}
```

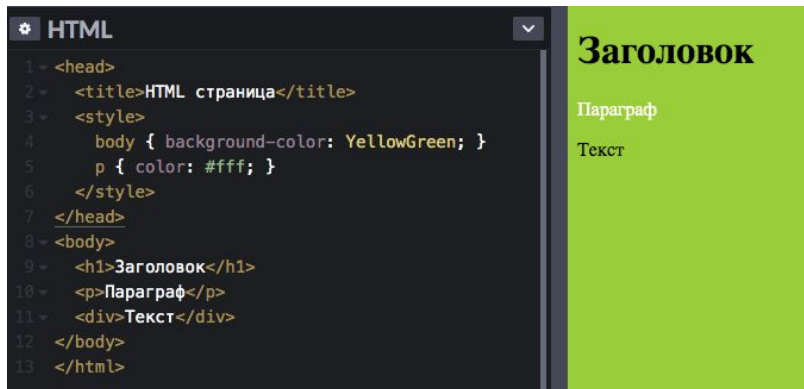
```
h1 {  
  
    border-bottom: 1px solid black;  
  
}
```

```
h1 {  
    color: gray;  
    font-family: sans-serif;  
    border-bottom: 1px solid black;  
}
```

Обычно дополнительные правила для элемента указываются в тех случаях, когда был задан один стиль сразу для нескольких элементов, но помимо этого необходимо добавить что-то ещё для определённого элемента:

```
h1, h2, h3 {  
  
    color: gray;  
  
    font-family: sans-serif;  
  
}  
  
h2 {  
  
    border-bottom: 1px solid black;  
  
}
```

Встроенные стили



Встроенные стили css прикрепляются напрямую к HTML-тегам значением атрибута style. Этот атрибут присутствует у всех HTML тегов, кроме тех, которые располагаются вне элемента **<body>**.

```
<p style="color: red">Текст сообщения</p>
```

В результате применения этого стиля, параграф будет показан красного цвета.

В атрибуте `style` может быть записано несколько значений стилей.

```
<p style="color: red; font-weight:bold">Текст сообщения</p>
```

Однако бросается в глаза, что так можно задавать только отдельные объявления конкретным элементам. Почему это неудобно?

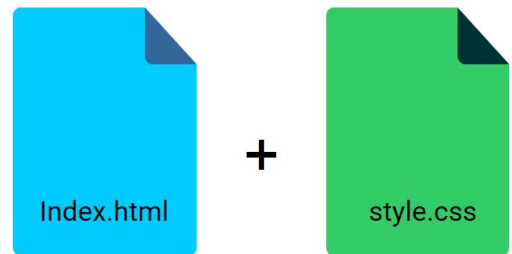
- Таблица стилей оказывается разбита и разбросана по всему HTML файлу, а это уничтожает большую часть преимуществ от ее применения
- Не получится написать общие определения вида «всем абзацам — 12 шрифт».
- Каждый элемент нужно стилизовать отдельно (это долго, нудно и получим массу повторяющегося кода);

Варианты подключения стилей



Существует 3 основных способа подключения или добавления CSS стилей к HTML документу, каждый из которых подходит для определенного круга задач.

- CSS стили для конкретного тега атрибутом `style`
- Подключение с помощью тега `link` через внешний файл стилей CSS
- Добавление CSS с помощью тега `style` в HTML файл



```
<head>  
  <link rel="stylesheet" href="style.css">  
</head>
```

С помощью атрибута style для тега

`<p style="color: red">Текст сообщения</p>`

```
HTML
1 <head>
2   <title>HTML страница</title>
3   <style>
4     body { background-color: YellowGreen; }
5     p { color: #fff; }
6   </style>
7 </head>
8 <body>
9   <h1>Заголовок</h1>
10  <p>Параграф</p>
11  <div>Текст</div>
12 </body>
13 </html>
```

Заголовок

Параграф

Текст

Подключение с помощью тега link через внешний файл стилей CSS

Подключение стилей CSS является использование элемента `link`, который позволяет подключать к HTML странице внешние файлы. Ссылка на внешний CSS файл помещается в контейнер `<head>` страницы:

```
<link rel="stylesheet" href="style.css" type="text/css"/>
```

В атрибуте `href` необходимо указать URL адрес файла, содержащего набор стилей CSS. Атрибуты `rel="stylesheet"` и `type="text/css"` указывают, что указанный файл является таблицей стиля в формате CSS.

Подключение с помощью тега link через внешний файл стилей CSS

Преимущества внешних CSS:

- Меньший размер страницы HTML и более чистая структура файла.
- Быстрая скорость загрузки.
- Для разных страниц может быть использован один и тот же .css файл.

Недостатки внешних CSS:

- Страница может некорректно отображаться до полной загрузки внешнего CSS.

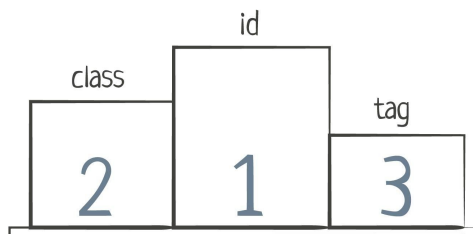
Добавление CSS с помощью тега style в HTML файл

Этот метод подойдет когда нужно вставить (определить) стили для группы уникальных элементов страницы, то есть, набор стилей, которые используются только в пределах одной страницы и не нужны для корректной работы остальных страниц сайта.

В любом месте областей `<head>` и `<body>` HTML документа используйте тег `<style>`, внутри которого поместите необходимые CSS правила.

```
<style type="text/css">
    h2 {
        color: red;
    }
</style>
```

Приоритеты выполнения стиля



Важной чертой CSS является то, что разные типы селекторов имеют разный приоритет. Если стили противоречат друг другу в разных селекторах, то вступает в дело принцип приоритета.

Разберём на примере. Создадим параграф с классом `red` и идентификатором `blue`

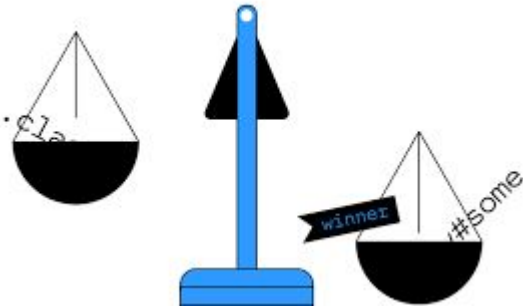
```
<p id="blue" class="red">Текст с классом и id</p>
```


Добавим противоречащие друг другу стили для класса, идентификатора и тега. Какого цвета будет параграф?

```
p {  
    color: black;  
}  
  
.red {  
    color: red;  
}  
  
#blue {  
    color: blue;  
}
```

Для определения приоритета можно использовать следующие правила, где каждому селектору задаётся его «вес»:

- Селектор по тегу: 1
- Селектор по классу и атрибуту: 10
- Селектор по ID: 100
- Стиль в атрибуте тега: 1000



Чтобы узнать, какой селектор будет иметь больший вес, нужно сложить все полученные значения. Например:

- Селектор `.paragraph` состоит из одного класса, а значит его вес — 10
- Селектор `.paragraph.color-primary` состоит из двух классов. Его вес — 20

<code>style=""</code>	1,0,0,0
<code>#id</code>	0,1,0,0
<code>.class</code>	0,0,1,0
<code>[attr=value]</code>	0,0,1,0
<code>LI</code>	0,0,0,1
<code>*</code>	0,0,0,0

Объявление !important

Если вы столкнулись с экстренным случаем и вам необходимо повысить значимость какого-либо свойства, можно добавить к нему объявление **!important**:

```
p {  
  
  color: red !important;  
  
  font-weight:bold  
  
}
```

Также **!important** перекрывает inline-стили. Слишком частое применение **!important** не приветствуется многими разработчиками. В основном, данное объявление принято использовать лишь тогда, когда конфликт стилей нельзя победить иными способами.