Рекомендуемая литература

1. Бен Фрейн. *HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств.* – СПб.: Питер, 2014. – 304 с.: ил.

2. Макфарланд Дэвид. *Большая книга CSS3.* 3-е изд. – СПб.: Питер, 2014. – 608 с.: ил. – (Серия «Бестселлеры O`Reilly»).

Интернет-ресурсы

www.w3.org
mva.microsoft.com – курс «Adding Style with CSS»
coursera.org

КАСКАДНЫЕ ТАБЛИЦЫ СТИЛЕЙ

CSS - Cascading Style Sheets

```
<FONT size=5 color=red>
  <I> Красный курсив размера 5 </I>
</FONT>
обычный текст
<FONT size=5 color=red>
  <I>опять красный курсив</I>
</FONT>
опять обычный текст
```

Типы стилей.

- Стиль автора.
- Стиль пользователя.
- Стиль браузера.

Преимущества стилей.

- Разграничение кода и оформления.
- Разное оформление для разных устройств.
- Расширенные по сравнению с HTML способы оформления элементов.
- Ускорение загрузки сайта.
- Единое стилевое оформление множества документов.
- Централизованное хранение.
- Реализация функций адаптивного дизайна.

Простейшее описание стиля.

```
<I style="font-size: 5; color: red; background-color: black">
   Это текст, к которому применен стиль.
</I>
```

Селекторы тегов (селекторы типов).

Селекторы тегов определяют какие элементы подлежат форматированию.

```
<STYLE> <!--
      Селектор1 { Блок Объявления CSS-правил }
      Селектор2 {
         атрибут_1: значение;
         атрибут_2: значение;
          ...
      }
--></STYLE>
```

```
   <STYLE>
       I {font-size: 18px; color: orange; background-color: navy}
   </STYLE>
Теперь весь <I>курсив</I> на странице будет выглядеть <I>так</I>,
кроме <I style="color: navy; background-color: orange">
 <FONT size=3px> специально отформатированных мест</FONT></I>.
```

Теперь весь *курсив* на странице будет выглядеть *так*,
кроме *специально отформатированных мест*.

Встраивание CSS в документ.

✶ Встроенные (внутренние) в документ таблицы стилей:

```
<HEAD>
     <STYLE type="text/css">   HTML 4.01
       CSS-правила
     </STYLE>
     <STYLE>                   HTML 5
       CSS-правила
     </STYLE>

</HEAD>
```

✶ Встраивание в теги.

```
<I style="font-size: 5; color: red">
   ...
</I>
```

✶ Присоединённые (внешние) таблицы стилей:

HTML 4.01:

```
<LINK rel="stylesheet" href="styles.css" type="text/css">
```

HTML 5:

```
<LINK rel="stylesheet" href="styles.css">
```

Файл *styles.css*:

```
H1 {
     color: #000080;
     font-size: 2em;
   }
P  { padding-left: 20px; }
```

✶ Импортирование таблиц стилей:

```
<STYLE type="text/css">
  @import url("имя файла");
  @import "имя файла";
</STYLE>
```

Файл *styles.css*:

```
@import url (myStyle.css)
@import "/style/print.css";
@import "/style/palm.css";
H1 {
     color: #000080;
     font-size: 2em;
```

```
    }
P   { padding-left: 20px; }
```

Что лучше/быстрее @import или LINK?
(www.stevesouders.com/blog/2009/04/09/dont-use-import)

Типы носителей.

- Атрибут *media* тегов *LINK* и *STYLE*

```
<HEAD>
 <!-- Во время презентации все H1 будут голубыми -->
 <STYLE type="text/css" media="projection">
    H1 { color: blue }
 </STYLE>
 <!-- При печати все объекты H1 будут центрироваться -->
 <STYLE type="text/css" media="print">
   H1 { text-align: center }
 </STYLE>
 <!-- Добавлен звуковой эффекты при голосовом выводе -->
 <STYLE type="text/css" media="aural">
   A { cue-before: uri(bell.aiff); cue-after: uri(dong.wav)}
 </STYLE>
</HEAD>
```

Тип носителя:

HTML 5:	*HTML 4.01:*
screen	tty
all	tv
print	projection
speech	handheld
	braille
	aural

Пример:

```
<LINK rel="stylesheet"  media="aural"
      href="aural.css"  type="text/css">
<LINK rel="stylesheet"  media="screen"
      href="screen.css" type="text/css">
<LINK rel="stylesheet"  media="print"
      href="print.css"  type="text/css">
<LINK rel="stylesheet"  href="techreport.css"
      type="text/css">
<STYLE media="screen, print" type="text/css">
       p.special { color: rgb(230, 100, 180) }
</STYLE>
```

- Эт-правила: **@import** и **@media**

```
@import url("имя файла") типы носителей;
@import "имя файла" типы носителей;
```

```
@media not|only тип-носителя and (характеристики) {
  CSS-правила;
}
```

Пример 1.

```
<style type="text/css">
  @import "style/main.css" screen;
     /* Стиль для вывода результата на монитор */
  @import "style/print.css" print, handheld;
     /* Стиль для печати и смартфона */
</style>
```

Пример 2.

```
<style type="text/css">
  @media screen { /* Стиль для отображения в браузере */
    H1 {
      background: #faf0e6;    /* Цвет текста */
      color: #a0522d;         /* Цвет текста */
    }
    H2 { color: #556b2f; }       /* Цвет текста */
    P  { margin-top: 0.5em; }    /* Отступ сверху */
  }
  @media print { /* Стиль для печати */
    H1, H2, P { color: black; } /* Черный цвет текста */
  }
</style>
```

Характеристики медиа-запросов:

```
width, height                          color
device-width, device-height            color-index
orientation: portrait | landscape      monochrome
aspect-ratio: 16/9                     resolution: dpi | dpcm
device-aspect-ratio                    scan: progressive | interlace
                                       grid: 0 | 1
```

Пример 1. Какой вид придадут стили содержимому? (в браузере).

```
body {
  background-color: grey;
}
@media screen and (max-width: 960px) {
  body { background-color: red; }
}
@media screen and (max-width: 768px) {
  body { background-color: orange; }
```

```
    }
    @media screen and (max-width: 550px) {
    body { background-color: yellow; }
    }
```

Пример 2.

```
<link rel="stylesheet"
  media="not screen and (orientation: portrait)"
  href="portrait-screen.css"
>
<link rel="stylesheet"
  media="screen and (orientation: portrait) and (min-width: 800px)"
  href="portrait-screen.css"
>
<link rel="stylesheet"
  media="screen and (orientation: portrait) and (min-width: 800px),
         projection" href="800wide-portrait-screen.css"
  href="portrait-screen.css"
>
```

Пример 3. Демо: разные стили на экране, проекторе и при печати (в браузере).

Задание стилей. Селекторы тегов.

```
<STYLE>
    I {font-size: 18px; color: orange; background-color: navy}
</STYLE>
Теперь весь <I>курсив</I> на странице будет выглядеть <I>так</I>,
кроме <I style="color: navy; background-color: orange">
<FONT size=3px> специально отформатированных мест</FONT></I>.
```

Теперь весь *курсив* на странице будет выглядеть *так*,
кроме *специально отформатированных мест*.

Контекстные селекторы:

```
<STYLE>
   I B {font-size: 18; color: white; background-color: black}
</STYLE>
Это <I>обычный курсив, а это
    <B>полужирный курсив </B> </I> с новыми <I> свойствами</I>.
```

Это *обычный курсив, а это*
полужирный курсив с новыми
свойствами.

Тег1 Тег2 { … } → *<Тег1>*
<Тег2> … </Тег2>
</Тег1>

Соседние элементы:

```
<STYLE type="text/css">
  B + I   { color: blue; }
  I + DEL { color: orange; }
</STYLE>
```

```
<B>фрукты</B> - это <I> яблоки </I>,
<I> груши </I>, а не <DEL> репа и
свёкла </DEL>.
```

фрукты - это *яблоки*, *груши*,
а не ~~репа и свёкла~~.

```
<B>фрукты</B> - это <I> яблоки,
груши </I>, а не <DEL> репа и свёкла
</DEL>.
```

фрукты - это *яблоки, груши*,
а не ~~репа и свёкла~~.

Селектор 1 + Селектор 2 { Описание правил стиля }

Последовательные элементы:

```
<STYLE type="text/css">
  B ~ I   { color: blue; }
  I ~ DEL { color: orange; }
</STYLE>
```

```
<B>фрукты</B> - это <I> яблоки </I>, <I> груши </I>, а не <DEL> репа и свёкла </DEL>.
```

фрукты - это *яблоки*, *груши*, а не ~~репа и свёкла~~.

```
<B>фрукты</B> - это <I> яблоки, груши </I>, а не <DEL> репа и свёкла </DEL>.
```

фрукты - это *яблоки, груши*, а не ~~репа и свёкла~~.

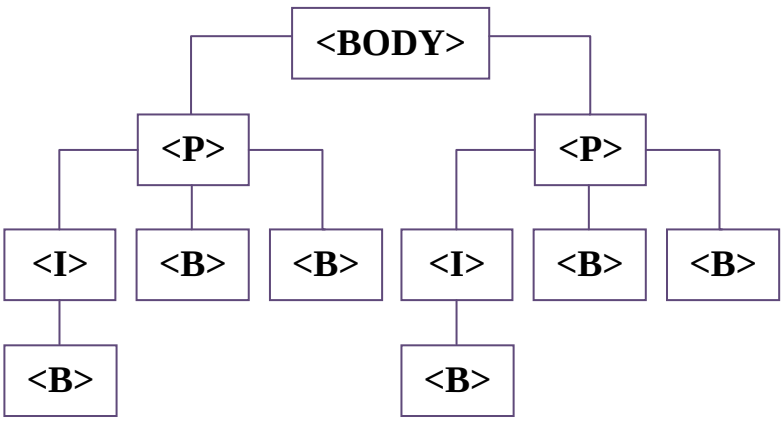
Селектор 1 ~ Селектор 2 { Описание правил стиля }

Дочерние селекторы:

HTML-разметка:

```
<BODY>
  <P>
    <I>
      <B>Фрукты:</B>
    </I>
    <B>Груши</B>
    <B>Яблоки</b>
  <P>
    <I>
      <B>Овощи:</B>
    </I>
    <B>Репа</B>
    <B>Свёкла</B>
  </P>
</BODY>
```

Дерево элементов:



```
<STYLE type="text/css">
   P > B { color: silver; }
</STYLE>
```

Фрукты: Груши Яблоки

Овощи: Репа Свёкла

Селектор 1 > Селектор 2 { Описание правил стиля }

Классы

Имя_дескриптора . Класс { CSS-правила }
. Класс { CSS-правила }

```
<STYLE>
   I.myStyle {
      font-size: 18; color: purple;
      background-color: pink
}
</STYLE>
<I>Книга – книгой</I>, а
<I class="myStyle">мозгами</I> двигай!
```

Книга – книгой, а *мозгами* двигай!

НО!!!

```
<DIV class="myStyle"> Книга – книгой, а мозгами двигай! </DIV>
```

Универсальное решение:

```
<STYLE>
   .mySt {
      color: darkred;
      background-color: lightblue
   }
   DIV {text-align: right}
</STYLE>

Какие часы <U class="mySt">чаще</U> показывают правильное
время – те, которые <I class="mySt">не работают</I>,
или те, которые <B class="mySt">отстают</B> на одну минуту?
<DIV class="mySt">Льюис Кэролл. "Задача о часах".</DIV>
```

Какие часы чаще показывают правильное время — те, которые *не работают*, или те, которые **отстают** на одну минуту?

Льюис Кэролл. "Задача о часах".

Мультиклассы

`<STYLE>`
`  .green {color: green;}`
`  .pink {background-color: pink}`
`  DIV {text-align: right}`
`</STYLE>`
	`<STYLE>`
	`  .green {color: green;}`
	`  .pink {background-color: pink}`
	`  .green.pink {text-align: right}`
	`</STYLE>`
	`!!! до IE10 не работает`

```
Какие часы <U class="pink">чаще</U> показывают
правильное время – те, которые
<I class="green">не работают</I>, или те, которые
<B class="green">отстают</B> на одну минуту?
<DIV class="pink green">Льюис Кэролл. "Задача о часах".</DIV>
```

Какие часы чаще показывают правильное время — те, которые *не работают*, или те, которые **отстают** на одну минуту?

Льюис Кэролл. "Задача о часах".

Пример 2. Конфликт стилей.

```
<style type="text/css">
   .layer1 { color: red; }
   .layer2 { color: blue; }
</style>
<p class="layer1">Текст красного цвета</p>
<p class="layer2">Текст синего цвета</p>
<p class="layer1 layer2">Какого цвета текст?</p>
```

Пример 3. Какого цвета текст?

```
<style type="text/css">
   .layer1 { color: red; }
   .layer2 { color: blue; }
   .layer1.layer2 { color: green; }
</style>
<p class="layer1">Текст красного цвета</p>
<p class="layer2">Текст синего цвета</p>
<p class="layer1 layer2"> Какого цвета текст?</p>
```

Пример 4. Какого цвета текст? (См. пример 3)

```
<style type="text/css">
   .layer1 { color: red; }
   .layer2 { color: blue; }
   .layer1  .layer2 { color: green; }
</style>
<p class="layer1">Текст красного цвета</p>
<p class="layer2">Текст синего цвета</p>
<p class="layer1 layer2"> Какого цвета текст?</p>
```

Идентификаторы.

Тег#Имя идентификатора { CSS-правила }

#Имя идентификатора { CSS-правила }

```
<STYLE>
   #p24 { background-color: orange }
   H6#f3 { text-decoration: underline }
</STYLE>
<H6 id="f3">Льюис Кэролл. "Задача о часах".</H6>
<SPAN id="p24">Какие часы чаще показывают правильное
время — те, которые не работают, или те, которые отстают
на одну минуту?</SPAN>
```

Льюис Кэролл. "Задача о часах".

Какие часы чаще показывают правильное время — те, которые не работают, или те, которые отстают на одну минуту?

Универсальный селектор.

**** { CSS-правила }***

```
* { margin: 0;
    padding: 0; }
```

«Особенности» универсального селектора:

1. Может вызвать «зависание» браузера.

2. «Непредсказуемый» эффект (в браузере):

```
* {
   display: block;
   border: 1px solid #c00;
}
```

3. Использование в комбинации с другими селекторами:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<style type="text/css">
```

```
      td  * {
         border: 2px solid purple; /* Параметры рамки */
         background: pink; /* Цвет фона */
      }
   </style>
   <form action="handler.php">
      В разных браузерах
   </form>
```

Совместное использование элементов группировки, классов, идентификаторов.

Пример 1. Что «покажет» браузер?

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
      http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<head>
  <style type="text/css">
    A {
      color: green; /* Зеленый цвет ссылок */
    }
    .menu {
     padding: 7px;      /* Поля вокруг текста */
     border: 1px solid darkblue; /* Параметры рамки */
     background: orange; /* Цвет фона */
    }
    .menu A {
     color: navy; /* Темно-синий цвет ссылок */
    }
  </style>
</head>
<body>
  <div class="menu">
    <a href="1.html">Планеты</a> |
    <a href="2.html">Звёзды</a> |
    <a href="3.html">Туманности</a>
  </div>
  <p><a href="text.html">Другие материалы</a></p>
</body>
```

```
</html>
```

Селекторы атрибутов.

- Выбор всех элементов с указанным атрибутом (*CSS2*)

[атрибут] { Описание правил стиля }
Селектор[атрибут] { Описание правил стиля }

```
<style type="text/css">
   Q         { font-style: italic; }
   Q[title]  { color: yellow; }
</style>
Текст, написанный <Q> курсивом </Q>
и <Q title="другой текст"> жёлтым курсивом </Q>.
```

- Все элементы с атрибутом, значение которого известно (*CSS2*)

[атрибут="значение"] { Описание правил стиля }
Селектор[атрибут="значение"] { Описание правил стиля }

```
<style type="text/css">
A[target="_blank"] {
  background: url(blank.png) 0 6px; /* Параметры фона */
}
</style>
<p><a href="1.html">Обычная ссылка</a> |
<a href="2.html" target="_blank"> Другая ссылка </a></p>
```

- Значение атрибута начинается с указанного значения (*CSS2*)

[атрибут|="значение"] { Описание правил стиля }
Селектор[атрибут|="значение"] { Описание правил стиля }

```
<style type="text/css">
  A[href|="http://"] { font-weight: bold }
  /* Жирное начертание ссылок, начинающихся с http:// */
</style>
```

- То же, что и |= (*CSS3*)

[атрибут^="значение"] { Описание правил стиля }

Селектор[атрибут^="значение"] { Описание правил стиля }

- Значение атрибута содержит указанную строку (***CSS2***)

[атрибут~="значение"] { Описание правил стиля }
Селектор[атрибут~="значение"] { Описание правил стиля }

```
<style type="text/css">
  [class~="block"] h3 { color: green; }
</style>
<div class="block tag">
  <h3>Зелёный заголовок</h3>
</div>
<h3>Обычный заголовок</h3>
```

- То же, что и **~=** (***CSS3***)

[атрибут*="значение"] { Описание правил стиля }
Селектор[атрибут*="значение"] { Описание правил стиля }

- Значение атрибута завершается указанной строкой (***CSS3***)

[атрибут$="значение"] { Описание правил стиля }
Селектор[атрибут$="значение"] { Описание правил стиля }

```
<style type="text/css">
  A[href$=".ru"] { font-weight: bold }
</style>
```

- Комбинация селекторов атрибутов

[атрибут1="значение1"][атрибут2="значение2"]
{ Описание правил стиля }
Селектор[атрибут1="значение1"][атрибут2="значение2"] {
Описание правил стиля
}

Псевдоклассы.

Селектор:Псевдокласс { CSS-правила }

```
A.menu:hover { color: green }
.menu A:hover { background: #fc0 }
:hover { font-weight: bolder }
```

Условное деление на:

- Псевдоклассы, определяющие состояние элементов
- Псевдоклассы, имеющие отношение к дереву документа
- Псевдоклассы, задающие язык текста

- Псевдоклассы, определяющие состояние элементов: распознают текущее состояние элемента и применяют стиль только для этого состояния.

1. :active (CSS1)
2. :link (CSS1)
3. :focus(CSS2)
4. :hover (CSS1)

```
<style type="text/css">
  TR:hover { background: #fc0; }
</style>
```

5. :visited (CSS1)

A:visited:hover – верно
A:link:visited – не верно

6. :target (CSS3)

```
<style>
  .tab div        { display: none; }   Скрыть элемент
  .tab div:target { display: block; }  Показать элемент
</style>
<div class="tab">
  <a href="#link1">Link 1</a>
  <div id="link1">
    <h3>Содержимое Link 1</h3>
    <p>Hello World!</p>
  </div>
```

7. :checked (CSS3)
8. :disabled (CSS3) :enabled (CSS3)
9. :required (CSS3) :optional (CSS3)

10. :read-only (CSS3) :read-write (CSS3)
11. :in-range (CSS3) :out-of-range (CSS3)

```
<style>
  input:in-range {
    border: 2px solid yellow;
  }
  input:out-of-range {
    border: 2px solid red;
  }
</style>
<input type="number" min="5" max="10" value="7">
```

12. :valid (CSS3) :invalid (CSS3)

- Псевдоклассы, имеющие отношение к дереву документа.

1. :first-child (CSS2)

```
<style type="text/css">
  P { text-indent: 1em; }        /* Отступ первой строки */
  P:first-child {    /*Для первого абзаца отступ убираем */
    text-indent: 0;
}
</style>
<body>
  <p>ПРИНЦЕССА или ТИГР?
  <p>- А что, если в обеих комнатах сидят тигры?
  <p>- Считай, не повезло.
  <p>- А если в обеих комнатах по красавице?
  <p>- Считай, подфартило.
</body>
```

ПРИНЦЕССА или ТИГР?

- А что, если в обеих комнатах сидят тигры?

- Считай, не повезло.

- А если в обеих комнатах по красавице?

- Считай, подфартило.

2. :first-of-type (CSS3)

P:first-child	P:first-of-type
{background-color: pink;}	{background-color: pink;}

```
<p>ПРИНЦЕССА или ТИГР?
<div>
  <br>- А что, если в обеих комнатах сидят тигры?
  <p>- Считай, не повезло.
  <p>- А если в обеих комнатах по красавице?
  <p>- Считай, подфартило.
</div>
```

ПРИНЦЕССА или ТИГР?

- А что, если в обеих комнатах сидят тигры?

- Считай, не повезло.

- А если в обеих комнатах по красавице?

- Считай, подфартило.

ПРИНЦЕССА или ТИГР?

- А что, если в обеих комнатах сидят тигры?

- Считай, не повезло.

- А если в обеих комнатах по красавице?

- Считай, подфартило.

3. `:last-child` (CSS3) `:last-of-type` (CSS3)
4. `:only-child` (CSS3) `:only-of-type` (CSS3)
5. `:empty` (CSS3)

Пример к п. 3-5.

```
*:only-child { border: 2px solid red; }
u:only-of-type { background-color: pink; }
div > *:last-child { color: green; }
div:last-child { text-align: right; }
u:last-of-type { font-weight: bolder; }
*:empty { border: 3px dashed orange; }
```

```
<div>
  «<i>Ни одно <b>ископаемое</b> животное не может
  быть несчастно в любви.</i> <br>
  <u>Устрица может быть несчастна в любви».</u><br>
  <u>Какое заключение из этого можно сделать?</u><br>
  <q>Устрица — не ископаемое животное.</q>
</div>  <br>  <div></div>  <br>
<div>
  <span><i>Льюис Кэрролл.</i></span>  <br>
<u>Инверсная силлогистика</u>
</div>
```

*«Ни одно **ископаемое** животное не может быть несчастно в любви.*
Устрица может быть несчастна в любви».
Какое заключение из этого можно сделать?
“Устрица — не ископаемое животное.”

Льюис Кэрролл.
Инверсная силлогистика

6. `:root (CSS3)`
7. `nth-правила(odd | even | an + b) (CSS3)`
   `:nth-child(n)` `:nth-last-child(n)`
   `:nth-of-type(n)` `:nth-last-of-type(n)`

Пример (в браузере)

```
*:nth-child(odd) { background-color: pink;}
li:nth-of-type(3n+2) {color: navy; font-weight: bolder;}
div:nth-last-of-type(3) {font-weight: bolder; }
```

```
<div>Логический лабиринт. (Рэймонд Смаллиан)</div>
<div>Ну, король был человеком слова: в одной из комнат - принцесса, в каждой же из остальных - либо тигр, либо вообще никого нет.</div>
<div>Таблички на дверях "говорили":</div>
<ol type="I">
  <li>Принцесса находится в комнате с нечетным номером
  <li>Эта комната пуста
  <li>Либо утверждение V истинно, либо утверждение VII ложно
  <li>Утверждение I ложно
  <li>Утверждение II или утверждение IV истинно
  <li>Утверждение III, ложно
  <li>В комнате I принцессы нет
  <li>В этой комнате сидит тигр, комната IX пуста
  <li>В этой комнате сидит тигр, и утверждение VI ложно
</ol>
<div>Узник задумался.</div>
```

Логический лабиринт. (Рэймонд Смаллиан)

Ну, король был человеком слова: в одной из комнат - принцесса, в каждой же из остальных - либо тигр, либо вообще никого нет.

Таблички на дверях "говорили":

I. Принцесса находится в комнате с нечетным номером
II. Эта комната пуста
III. Либо утверждение V истинно, либо утверждение VII ложно
IV. Утверждение I ложно
V. Утверждение II или утверждение IV истинно
VI. Утверждение III, ложно
VII. В комнате I принцессы нет
VIII. В этой комнате сидит тигр, комната IX пуста
IX. В этой комнате сидит тигр, и утверждение VI ложно

Узник задумался.

- Псевдоклассы, задающие язык текста.

1. :lang(язык) (CSS2)

```
<style type="text/css">
   q:lang(de) {
     quotes: "\201E" "\201C";
   } /*кавычки для немецкого языка*/
   q:lang(fr), q:lang(ru) {
     quotes: "\00AB" "\00BB";
   } /*кавычки для русского и французского языка*/
</style>
<p>Цитата на: <q lang="fr"> французском языке. </q></p>
<p>Цитата на: <q lang="de"> немецком языке. </q></p>
```

Псевдоэлементы.

Селектор::Псевдоэлемент { Описание правил стиля }

```
.foo::first-letter { color: red }
.foo::first-line {font-style: italic}
```

1. ::first-letter (CSS1)
2. ::first-line (CSS1)
3. ::after (CSS2)
4. ::before (CSS2)
5. ::selection

Пример 1.

```
UL {
  padding-left: 0; /* Убираем отступ слева */
  list-style-type: none; /* Прячем маркеры списка */
}
LI::before {content: "*";}
   <ul>
     <li>Метод простых итераций</li>
     <li>Метод случайных чисел</li>
     <li>Метод золотого сечения</li>
   </ul>
```

Пример 2.

```
div::after  { content: attr(data-end); }
div::before { content: attr(data-mytitle); }
   <div data-mytitle="Привет " data-end=" !">Вася</div>
```

Привет Вася !

Правила применения стилей.

Повторение свойств.

```
P { color: green; }
P { color: red; }
```

Группировка и наследование.

1. Группировка селекторов:

```
H1 {font-family: Arial}
H2 {font-family: Arial}
H3 {font-family: Arial}
```

```
H1, H2, H3 {font-family: Arial}
```

2. Группировка определений:

```
H1 {font-family: Arial}
H1 {font-size: 14pt}
H1 {font-weight: bold}
```

```
H1{font-family: Arial;
   font-size: 14pt;
   font-weight: bold}
H1{font: bold 14pt Arial}
```

3. Наследование.

```
<P>…<EM>единый стиль</EM>…</P>
```

Значения различных стилевых свойств.

- Строка.

```
'Гостиница "Турист"'
"Гостиница 'Турист'"
"Гостиница \"Турист\""
'Гостиница \'Турист\''

a[title="a not s\
o very long title"] {/*...*/}
a[title="a not so very long title"] {/*...*/}
```

- Число: `.7` и `0.7` равнозначны.
- Проценты: 56.8%, НО не всегда.
- Размер.

Относительные единицы:

`em` – 1em=100% браузера.
`rem` – 1rem=100% родителя.
`ex` – 1ex = высота «x»
`px` – пикселы;
`%` – проценты;

`vh, vw` – 1% высоты/ширины области отображения;
`vmin` – 1vmin=min(1vh, vw);
`vmax` – 1vmax=max(1vh, vw);
`ch` – 1ch = ширина «0».

Абсолютные единицы:

`in` – дюйм (1in = 2,54cm);
`cm` – сантиметр;
mm – миллиметр;
`pt` – пункт (1pt = 1/72 in);

`pc` – пика (1pc = 12 in);
`px` – пиксел (1px = 1/96 in)
`q` – 1q = 1/40 cm

- Цвет.

- *#RRGGBB* или *#RGB* (`#fe0 = #ffee00`)

- по названию;

Примеры названий цветов и их RGB-значений.

Named	Numeric	Color name	Hex rgb	Decimal
		black	#000000	0,0,0
		silver	#C0C0C0	192,192,192
		gray	#808080	128,128,128
		white	#FFFFFF	255,255,255
		maroon	#800000	128,0,0
		red	#FF0000	255,0,0
		purple	#800080	128,0,128
		fuchsia	#FF00FF	255,0,255
		green	#008000	0,128,0
		lime	#00FF00	0,255,0
		olive	#808000	128,128,0
		yellow	#FFFF00	255,255,0
		navy	#000080	0,0,128
		blue	#0000FF	0,0,255
		teal	#008080	0,128,128
		aqua	#00FFFF	0,255,255

Named	Numeric	Color name	Hex rgb	Decimal
		aliceblue	#F0F8FF	240,248,255
		antiquewhite	#FAEBD7	250,235,215
		aqua	#00FFFF	0,255,255
		aquamarine	#7FFFD4	127,255,212
		lightcoral	#F08080	240,128,128
		lightcyan	#E0FFFF	224,255,255
		lightgoldenrodyellow	#FAFAD2	250,250,210
		lightgray	#D3D3D3	211,211,211
		lightgreen	#90EE90	144,238,144
		lightgrey	#D3D3D3	211,211,211
		lightpink	#FFB6C1	255,182,193
		lightsalmon	#FFA07A	255,160,122
		lightseagreen	#20B2AA	32,178,170
		lightskyblue	#87CEFA	135,206,250
		lightslategray	#778899	119,136,153

Доступ к цветовой схеме системы:

ActiveBorder	*ButtonText*	*InactiveCaptionText*	*ThreeDFace*
ActiveCaption	*CaptionText*	*InfoBackground*	*ThreeDHighlight*
AppWorkspace	*GrayText*	*InfoText*	*ThreeDLightShadow*
Background	*Highlight*	*Menu*	*ThreeDShadow*
ButtonFace	*HighlightText*	*MenuText*	*Window*
ButtonHighlight	*InactiveBorder*	*Scrollbar*	*WindowFrame*
ButtonShadow	*InactiveCaption*	*ThreeDDarkShadow*	*WindowText*

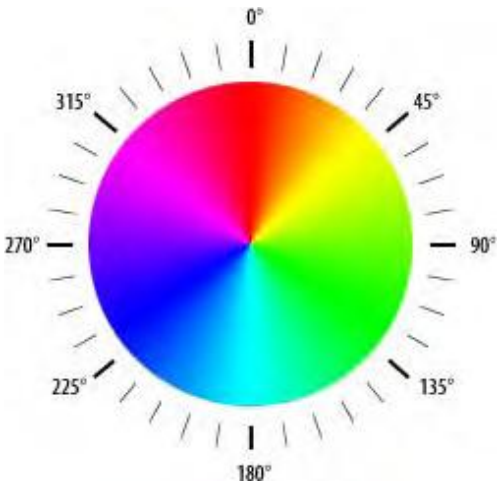
- формат RGB: *rgb*(255, 128, 128) или *rgb*(100%, 50%, 50%).
- формат RGBA: альфа-канал (0 – полная прозрачности, 1 – непрозрачность) (CSS3).

```
H2 {
   background-color: rgb(214,86,43);
   color: rgba(255,255,255,.9);
}
```

- формат HSL. (CSS3)

Оттенок (0..359): 0° – красный цвет, 120° – зелёный, 240° – синий

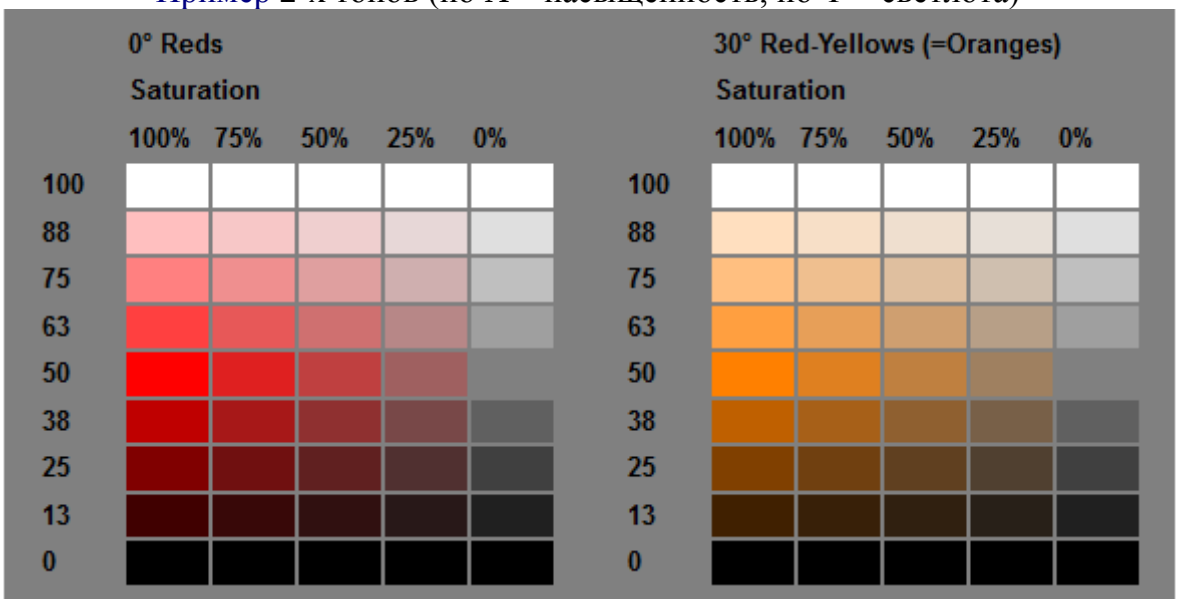
Насыщенность (0..100%): 0% – отсутствие цвета и оттенок серого



Светлота (0..100%) 0% - чёрные цвета, 100% - белые цвета.

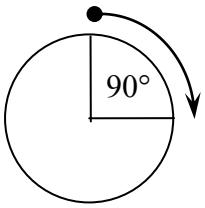
```
orange #ffa500 rgb(255,165,0) hsl(38.8,100%,50%) Оранжевый
yellow #ff0    rgb(255,255,0) hsl(60,100%,50%)   Жёлтый
olive  #808000 rgb(128,128,0) hsl(60,100%,25%)   Оливковый
```

Пример 2-х тонов (по X – насыщенность, по Y – светлота)



- формат HSLA – альфа-канал (0 – полная прозрачности, 1 – непрозрачность) (CSS3).

```
DIV {
   background-color: hsl(60,100%,25%);
   color: hsla(120,100%,50%,0.1);
}
```



- Градусы.
  deg – градусы (360 deg – полный круг).
  grad – грады (gradians, gons, grades). 400 grad = полный круг.
  rad – радианы. 2π rad = полный круг, м.б. дробными.
  turn – поворот. 1 turn = полный круг, м.б. дробными.

- Время.
  s – секунды.
  ms – миллисекунды (1s = 1000ms).

- Частота.
  Hz – Герцы.
  kHz – килоГерцы (1kH = 1000 H).

- Разрешение.
  dpi – точек (dots) на дюйм.
  dpcm – точек на сантиметр.
  dppx – точек на пиксел.

- Адреса.

```
url(относительный или абсолютный адрес файла)
url("относительный или абсолютный адрес файла")
url('относительный или абсолютный адрес файла')
```

- Ключевые слова.

Правильно: `P { text-align: right; }`

Неверно: `P { text-align: "right"; }`

`inherit` – значение свойства родителя

`initial` – «возврат» к начальному значению

- Математическое выражение `calc()`.

Для величин *length*, *frequency*, *angle*, *time*, *number* и *integer*.

```
section {
  float: left;
  width: calc(100%/3 - 2*1em - 2*1px);
}
p { margin: calc(1rem - 2px) calc(1rem - 1px); }
:root {  font-size: calc(100vw / 40); }
.foo {
  background: url(top.png);
  background-position: calc(50% + 20px) calc(50% + 20px), 50% 50%;
}
```

НО неверно:

```
calc(5px - 5px + 10s)
calc(0 * 5px + 10s)
width(5px – 10px)
```

- Математическое выражение `toogle()`.

```
em { font-style: toggle(italic, normal); }
ul { list-style-type: toggle(disc, circle, square, box); }
margin: toggle(1px 2px, 3px 4px, 1px 5px);
margin-top:    toggle(1px, 3px, 1px);
margin-right:  toggle(2px, 4px, 5px);
margin-bottom: toggle(1px, 3px, 1px);
margin-left:   toggle(2px, 4px, 5px);
```

- Математическое выражение `attr()`. (пример на с.23)

Приоритет задания стилей.
Увеличение приоритета:

- ↓ Стиль браузера (стили по умолчанию).
- ↓ Связная таблица стилей (LINK).
- ↓ Импортируемая таблица стилей (@import).
- ↓ Правила из раздела <STYLE>.
- ↓ Правила с CLASS в качестве селектора.
- ↓ Правила с ID в качестве селектора.
- ↓ Встроенное в тег HTML правило.

Правила с `!important`:

- ↓ Стиль браузера (стили по умолчанию).
- ↓ Стиль пользователя.
- ↓ Стиль автора.
- ↓ Стиль автора с добавлением `!important`.
- ↓ Стиль пользователя с добавлением `!important`.

Свойство: значение !important;

***Специфичность селектора*.**

- # = 100
- . и : = 10
- селектор тега и :: = 1;
- атрибут *style* = 1000
- `!important`

Какова специфичность селекторов?

```
* {}
li {}
li:first-line {}
ul li {}
ul ol+li {}
ul li.red {}
li.red.level {}
p#t34 {}
#content #wrap {}
```

Пример 1. Что покажет браузер?

```
<style type="text/css">
  #menu ul li {
    color: green;
  }
  .two { color: red; }
</style>
<div id="menu">
  <ul>
```

```
    <li>Первый</li>
    <li class="two">Второй</li>
    <li>Третий</li>
  </ul>
</div>


/* 1. Понижаем специфичность первого селектора */
   ul li {...} /* Убираем идентификатор */
   .two {...}
/* 2. Повышаем специфичность второго селектора */
   #menu ul li {...}
   #menu .two {...} /* Добавляем идентификатор */
/* 3. Повышаем приоритет второго стиля */
   #menu ul li {...}
   .two { color: red !important; } /* Добавляем !important */
```

Пример 2. Что покажет браузер?

```
<style type="text/css">
  #A, .a {
    border: none;
    background: aqua;
    color: olive;
  }
  .b {
    border: 1px solid navy;
    color: navy;
  }
</style>
<p id="A" class="b">Стиль идентификатора</p>
<p class="a b">Стиль классов a и b</p>
<p class="b">Стиль класса b</p>
```

--ИмяЗначения
var(--ИмяЗначения [, допЗначения])

Пример 1.
```
:root {
  --main-color: #06c;
  --accent-color: #006;
}
#f34 h1 { color: var(--main-color); }
```

Пример 2.
```
:root { --color: blue; }
div { --color: green; }
#alert { --color: red; }
* { color: var(--color); }
    <p>Синие буквы – унаследовано от элемента root!</p>
    <div>Зелёные буквы!</div>
    <div id='alert'>
      Красные буквы!
      <p>Тоже красные буквы – унаследовано от div!</p>
    </div>
```

Пример 3.
```
:root {
  --main-color: #c06;
  --my-background: linear-gradient(to top, var(--main-color), white);
  background: var(--my-background);
}
```
НО неверно!
```
:root {
  --one: calc(var(--two) + 20px);
  --two: calc(var(--one) - 20px);
}
```

Пример 4. НЕВЕРНО!
```
.f1 {
  --side: margin-top;
  var(--side): 20px;
}
.f2 {
  --looks-valid: 20px;
  background-color: var(--looks-valid);
}
.f3 {
  --gap: 20;
```

```
  margin-top: var(--gap)px;           // НЕВЕРНО!
  margin-top: calc(var(--gap) * 1px); // ВЕРНО!
}
```

Пример 5.

```
.component .header {
  color: var(--header-color, blue);
}
.component .text {
  color: var(--text-color, black);
}
.component {
  --text-color: #080;
  /* значение header-color не определено, поэтому
     вызов var(--header-color, blue) вернёт blue */
}
```

API JavaScript: `getPropertyValue`(ПЗ_in_camel-cased_form)

Счетчики CSS2.1.

`counter-reset` – сброс счётчика;
`counter-increment` – изменение значения счётчика;
`counter()` или ~~`counters`~~`()` – ф-ции доступа к счетчику.

Пример 1.

```
body {
  counter-reset: section;
}
h3::before {
  counter-increment: section;
  content: "Section " counter(section) ": ";
}
<h3>Introduction</h3>
<h3>Body</h3>
<h3>Conclusion</h3>
```

Section 1: Introduction

Section 2: Body

Section 3: Conclusion

Пример 2. (в браузере)

```
ol {
  counter-reset: section;
  list-style-type: none;
}
li::before {
  counter-increment: section;
  content: counter(section) ". ";
}
<ol>
  <li>item</li>          <!-- 1     -->
  <li>item               <!-- 2     -->
    <ol>
      <li>item</li>      <!-- 2.1   -->
      <li>item</li>      <!-- 2.2   -->
      <li>item           <!-- 2.3   -->
        <ol>
          <li>item</li>  <!-- 2.3.1 -->
          <li>item</li>  <!-- 2.3.2 -->
        </ol>
        <ol>
          <li>item</li>  <!-- 2.3.1 -->
          <li>item</li>  <!-- 2.3.2 -->
          <li>item</li>  <!-- 2.3.3 -->
        </ol>
      </li>
      <li>item</li>      <!-- 2.4   -->
    </ol>
  </li>
  <li>item</li>          <!-- 3     -->
  <li>item</li>          <!-- 4     -->
</ol>
<ol>
  <li>item</li>          <!-- 1     -->
  <li>item</li>          <!-- 2     -->
</ol>
```

Ответы

Пример 2 (назад)