Cascading Style Sheets

Данильченко Анна Александровна

Преподаватель кафедры программного обеспечения систем ЖГТУ

Cascading Style Sheets

- *CSS (каскадные листы/таблицы стилей)* — это язык для описания стилей, которые задают внешний вид документов, написанных при помощи языков разметки.
- *CSS* позволяет устанавливать цвета, шрифты, отступы, фоны, размеры, управлять местоположением (позиционированием) и обтеканием элементов, реализовывать различные оформительские решения.

Идея использования HTML совместно с CSS

- **Разделение структуры и оформления документа.**

HTML используется для описания **логической** структуры документа (выделения заголовков, подзаголовков, абзацев, таблиц, гиперссылок и других базовых логических элементов)

CSS описывает **физическое** форматирование документа (как должен выглядеть тот или иной логический элемент документа)

Разделение оформления и структуры документа дает такие преимущества:

- возможность параллельной разработки/модификации документа и его оформления/дизайна.
- расширенные возможности по сравнению с HTML;
- возможность одновременного изменения внешнего вида множества документов при помощи одной таблицы стилей;
- возможность установки различного форматирования для различных носителей информации (экран, печать и т. п.).

Уровни CSS

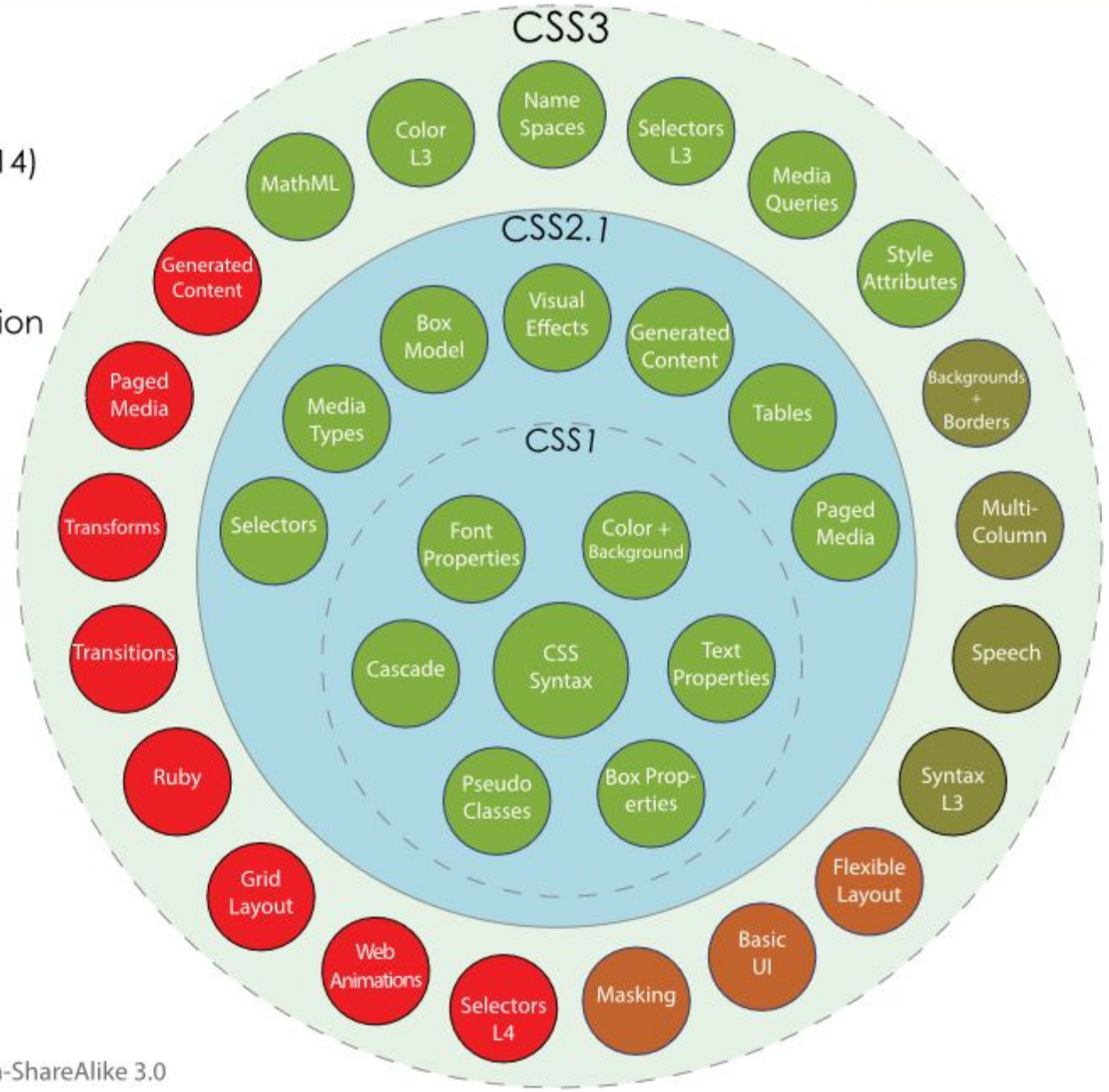
ПРИ РАЗРАБОТКЕ СТАНДАРТА CSS КОНСОРЦИУМ ВСЕМИРНОЙ ПАУТИНЫ ПРИНЯЛ РЕШЕНИЕ КЛАССИФИЦИРОВАТЬ НОВЫЕ СТАНДАРТЫ CSS НЕ ПО ВЕРСИЯМ, КАК ПРИНЯТО В РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, А ПО УРОВНЯМ.

Уровень CSS	Статус документа W3C	Дата принятия документа
CSS1 (уровень1)	рекомендация	17 декабря 1996 г., отредактирована 11 апреля 2008 г.
CSS2 (уровень 2)	рекомендация	12 мая 1998 г.
CSS2.1 (уровень 2, редакция 1)	кандидат на рекомендацию	19 июля 2007 г.
CSS3 (уровень 3)	в стадии разработки	

CSS3

Taxonomy & Status (October 2014)

- W3C Recommendation
- Candidate Recommendation
- Last Call
- Working Draft
- Obsolete or inactive



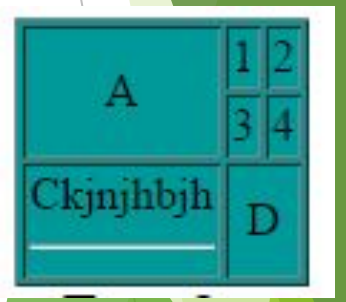
Способы включения CSS в HTML

1. Использование внешних таблиц стилей
2. Использование внутренних таблиц стилей
3. Использование встраиваемых стилей

Встраиваемые стили

Описание стиля располагается непосредственно внутри тега элемента, который описывается. Это делается с помощью параметра STYLE

```
<TABLE BORDER style="background:#099">
```

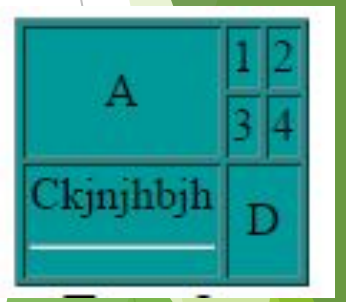


Этот метод нежелателен, он приводит к потере одного из основных преимуществ CSS – возможности отделения информации от описания оформления информации.

Внутренние таблицы стилей

Описание стилей располагается в коде Web-странички, внутри тега <STYLE type="text/css">... </STYLE>. Тег <STYLE> размещается внутри контейнера HEAD

```
<style>
table{background :#099;}
</style>
```

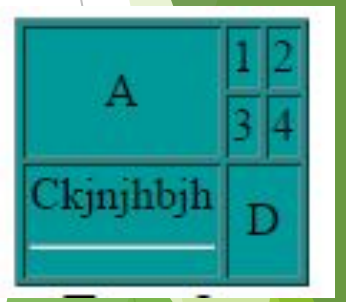


В этом случае описанные стили можно использовать для элементов, располагающихся в пределах странички.

Внешние таблицы стилей

Информация о стилях располагается в отдельном файле (*.css). Это имеет смысл в случае, если планируется применять эти стили к большему, чем одна, количеству страниц.

```
table{background :#099;}
```



Подключение файла *.css

<link rel="stylesheet" type="text/css" href="style2.css">

example.html

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="mystyles,css"/>
  </head>
  <body>
    содержание страницы HTML-документа
  </body>
</html>
```

Ссылка может быть как на «локальную» страницу стилей, созданную специально для этого документа, так и на «глобальную», хранящуюся в сети Интернет.

example1.html

```
<head>
  <link rel="stylesheet" type="text/css"
        href="http://www.google.com/uds/css/gsearch.css" />
</head>
```

MIME-типы

Internet media type (MIME – Multipurpose Internet Mail/Media Extension)

`application/javascript`	Текст на JavaScript (обычно .js)
`application/octet-stream`	Двоичная программа (обычно .exe)
`image/jpeg`	Картинка в формате JPEG
`image/gif`	Картинка в формате GIF
`image/vnd.microsoft.icon`	Иконка в формате Microsoft
`audio/mpeg`	Звуковой файл (обычно .mp3)
`text/plain`	Просто текст (например, .txt)
`text/html`	Текст на языке HTML
`text/css`	Каскадная таблица стилей (.css)
`video/mp4`	Кино в формате MP4

Подробнее см. http://ru.wikipedia.org/wiki/MIME

Базовый синтаксис

- **Стиль** – это набор параметров, задающих представление некоторого элемента веб-страницы.
- **Селектор** – это имя стиля.

Стили описываются в такой форме:

```
селектор {
        свойство1: значение1;
        свойство2: значение2;
        …
        свойствоN: значениеN;
}
```

Пример:

```
.footer {
   background: #960869;
}
```

Синтаксис CSS

Типы селекторов

- В качестве селекторов (имен стилей) могут использоваться:

  - универсальный селектор
  - теги
  - классы, определяемые пользователем
  - идентификаторы, определяемые пользователем;

Типы селекторов

1. Универсальный селектор
2. Селектор тегов
3. Селектор атрибутов
4. Селектор по классу
5. Селектор по идентификатору
6. Контекстный селектор
7. Дочерний селектор
8. Соседний селектор

Универсальный селектор
один стиль для всех элементов веб-страницы, например, задать шрифт или начертание текста

- * { Описание правил стиля }
- * { font-family: Arial, Verdana, sans-serif; /* *Рубленый шрифт для текста* */
- font-size: 96%; /* Размер текста */ }

Селектор тегов

В качестве селектора может выступать любой тег HTML, для которого определяются правила форматирования, такие как: цвет, фон, размер и т. д.

Тег { свойство1: значение; свойство2: значение; ... }

Пример:

```
P { text-align: justify; /* Выравнивание по ширине */
 color: green; /* Зеленый цвет текста */
 }
```

Селектор атрибутов

Устанавливает стиль для элемента, если задан специфичный атрибут тега.

[атрибут] { Описание правил стиля }

Селектор[атрибут] { Описание правил стиля }

A[href^="http://"] - начинается с определённого текста

A[href$=".ru"] - оканчивается определённым текстом

A[href*="htmlbook"] - содержит указанный текст

[class~="block"] - Одно из нескольких значений атрибута

DIV[class|="block"] - Дефис в значении атрибута
(**<div** class="block-menu-therm">)

CSS

Пример:

```
A[href$=".ru"] { /* Если ссылка заканчивается на .ru */
 background: url(images/ru.png) no-repeat 0 6px; /* Добавляем фоновый
    рисунок */
 padding-left: 12px; /* Смещаем текст вправо */ }
 A[href$=".com"] { /* Если ссылка заканчивается на .com */
background: url(images/com.png) no-repeat 0 6px;
 padding-left: 12px; }
```

HTML

```
<a href="http://www.yandex.com">Yandex.Com</a> |
<a href="http://www.yandex.ru">Yandex.Ru</a>
```

Селектор по классу

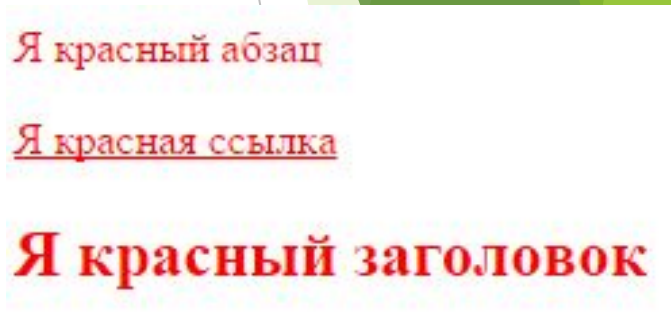
к элементам страницы добавляем слово class=“name” и задаем стили для класса

Пример HTML

```
<p class="red">Я красный абзац</p>
<a class="red" href="#">Я красная ссылка</a>
<h2 class="red">Я красный заголовок</h2>
```

CSS

```
.red{
    color:#F00;}
}
```



Селектор по идентификатору

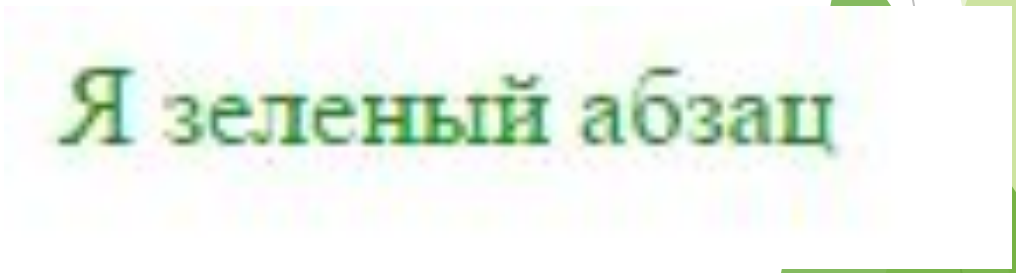
задаем элементу страницы уникальный идентификатор

Пример HTML

```
<p id="green">Я зеленый абзац</p>
```

CSS

```
#green{
    color:#090;
}
```



Идентификаторы или классы?????

Идентификаторы	Классы
В коде документа каждый идентификатор уникален и должен быть включён лишь один раз. Имя идентификатора чувствительно к регистру. Через метод getElementById можно получить доступ к элементу по его идентификатору и изменить свойства элемента. Стиль для идентификатора имеет приоритет выше, чем у классов.	Классы могут использоваться в коде неоднократно. Имена классов чувствительны к регистру. Классы можно комбинировать между собой, добавляя несколько классов к одному тегу.

Контекстный селектор

При создании веб-страницы часто приходится вкладывать одни теги внутрь других. Чтобы стили для этих тегов использовались корректно, помогут селекторы, которые работают только в определённом контексте. Например, задать стиль для тега <b> только когда он располагается внутри контейнера <p>.

Контекстный селектор состоит из простых селекторов разделенных пробелом.

Тег1 Тег2 { … }

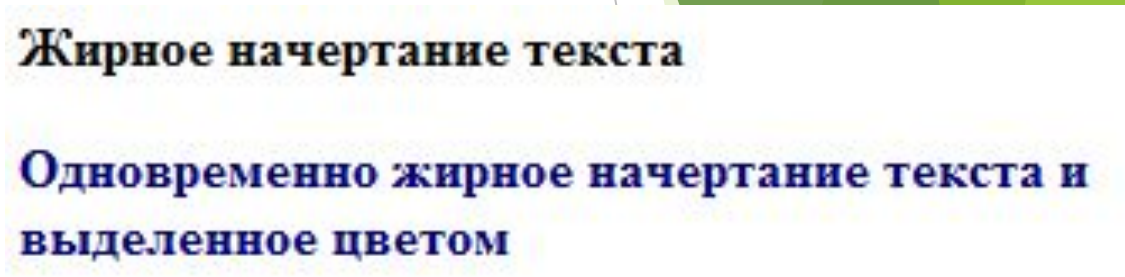
Контекстный селектор

Пример HTML

```
<div>
<b>Жирное начертание текста</b>
</div>
 <p><b>Одновременно жирное начертание текста и выделенное цветом</b>
</p>
```

CSS

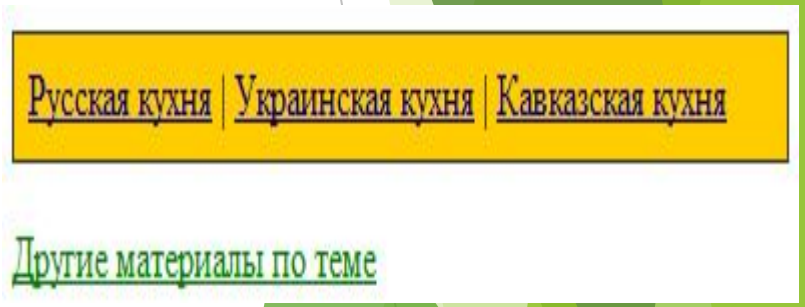
```
P B { font-family: Times, serif; /* Семейство шрифта */
color: navy; /* Синий цвет текста */ }
```



Контекстный селектор

HTML

```
<div class="menu">
 <a href="http://htmlbook.ru/1.html">Русская кухня</a> |
    <a href="http://htmlbook.ru/2.html">Украинская кухня</a> |
    <a href="http://htmlbook.ru/3.html">Кавказская кухня</a>
</div>
 <p>
<a href="http://htmlbook.ru/text.html">Другие материалы по теме</a>
</p>
```



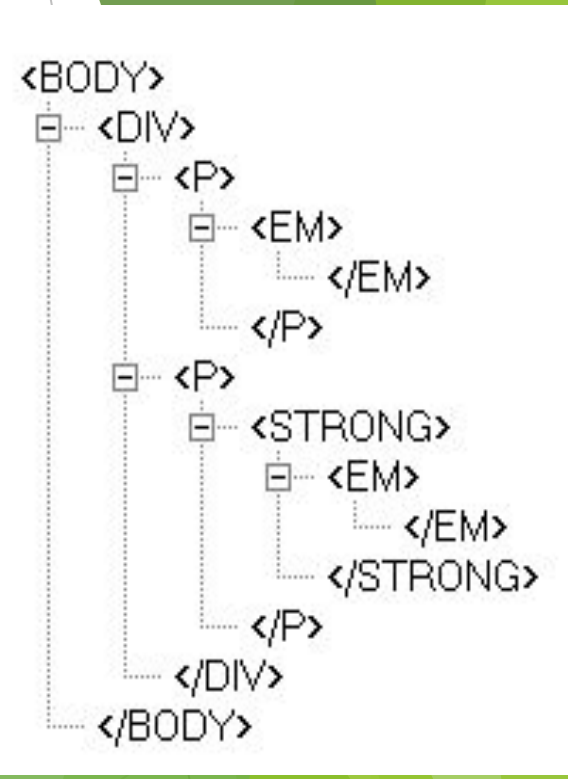
CSS

```
A { color: green; /* Зеленый цвет текста для всех ссылок */ }

 .menu { padding: 7px; /* Поля вокруг текста */
border: 1px solid #333; /* Параметры рамки */
 background: #fc0; /* Цвет фона */ }

 .menu A { color: navy; /* Темно-синий цвет ссылок */ }
```

Дочерний селектор

Дочерним называется элемент, который непосредственно располагается внутри родительского элемента

```
body
<div class="main">
 <p>
<em>Lorem ipsum dolor sit amet</em>, consectetuer adipiscing elit, sed diem
    nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
    volutpat.</em></p>
 <p>
<strong><em>Ut wisis enim ad minim veniam</em>
</strong>
, quis nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
    commodo consequat.
</p>
</div>
</body>
```



Дочерний селектор

Селектор 1 > Селектор 2 { Описание правил стиля }

HTML

** **

<p>

****Lorem ipsum dolor sit amet****

, consectetuer adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet

aliguam erat volutpat. **> **dolore magna****

</p

** **

CSS

DIV I { /* Контекстный селектор */ color: green; /* Зеленый цвет текста */ }

P > **I** { /* Дочерний селектор */ color: red; /* Красный цвет текста */ }

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet *dolore magna* aliguam erat volutpat.

Соседний селектор

Соседними называются элементы веб-страницы, когда они следуют непосредственно друг за другом в коде документа.

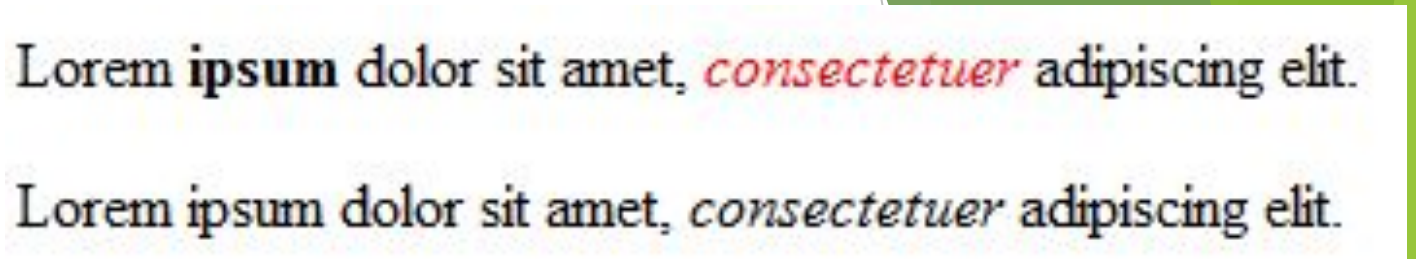
E + F { Описание правил стиля }

Стиль при такой записи применяется к элементу F, но только в том случае, если он является соседним для элемента E и следует сразу после него.

Lorem ipsum dolor sit amet. - дочерний

Lorem ipsum dolor <var>sit</var> amet. - соседний

Соседний селектор



Пример HTML

```
<p>Lorem <b>ipsum </b> dolor sit amet, <i>consectetuer</i>
   adipiscing elit.
</p>
<p>Lorem ipsum dolor sit amet, <i>consectetuer</i> adipiscing elit.
</p>
```

CSS

```
b + i { color: red; /* Красный цвет текста */ }
```

Псевдоклассы

- :first-child применяет стилевое оформление к первому дочернему элементу своего родителя.

```
tr:first-child{
    background-color:#06F;}
```

:nth-child используется для добавления стиля к элементам на основе нумерации в дереве элементов.

```
nav li:nth-child(odd){
        background-color:#CCC;}
```

odd	Все нечетные номера элементов.
even	Все четные номера элементов.
число	Порядковый номер дочернего элемента относительно своего родителя. Нумерация начинается с 1, это будет первый элемент в списке.

last-child задает стилевое оформление последнего элемента своего родителя

LI:nth-child(3n+3) - Он выбирает каждый третий элемент внутри маркированного списка: это 3-й, 6-й, 9-й, 12-й и т.д.

Псевдоэлементы :before и :after

Псевдоэлемент :before применяется для отображения желаемого контента до содержимого элемента, к которому он добавляется. Работает совместно со свойством content.

:before наследует стиль от элемента, к которому он добавляется.

after
Псевдоэлемент, который используется для вывода желаемого текста после содержимого элемента, к которому он добавляется.

Свойство 'Свойство 'content'

Значение:	[<строка> | <<uri> | <<счетчик> |
Начальное значение:	пустая строка
Область применения:	псевдоэлементы :before и :after
Наследование:	нет
Процентное задание:	нет
Устройства:	все

<строка>
Текстовое содержимое

<uri>
Значением является URI,
обозначающий внешний ресурс.

<счетчик>
Счетчики могут задаваться с помощью двух различных функций
'counter()' или 'counters()'.

Пример

1. one
    1.1. one-one
        1.1.1. one-one-one
        1.1.2. one-one-two
        1.1.3. one-one-three
    1.2. one-two
    1.3. one-three
2. two
3. three
    3.1. three-one
    3.2. three-two
    3.3. three-three

```
ol>li{
display:block
}

ol>li:before{
content:counters(item,".") ". ";
counter-increment:item
}
ol{
counter-reset:item
}
```

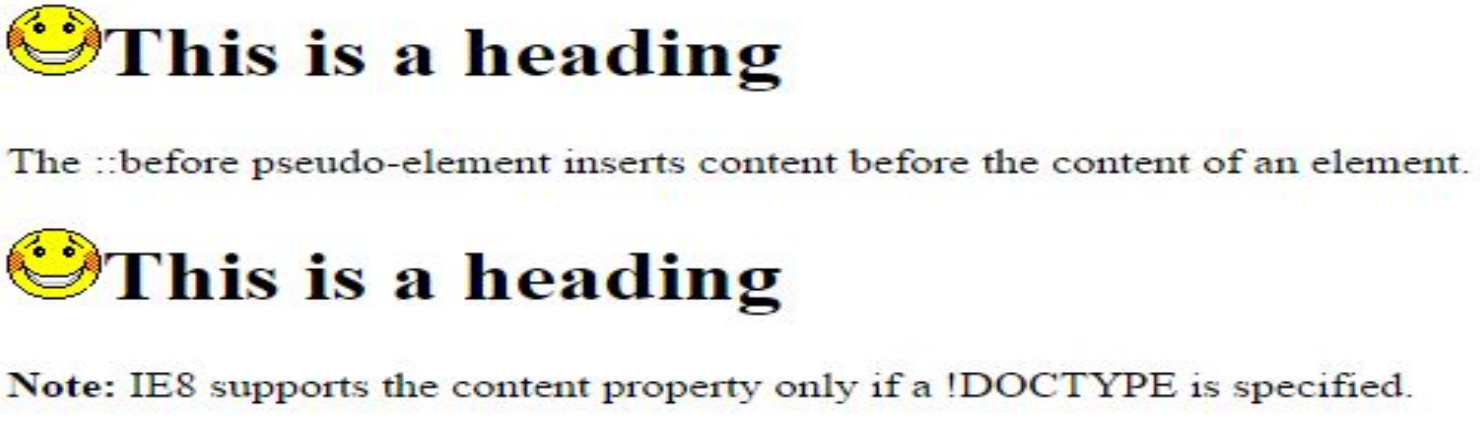
```
<ol>
<li>one
    <ol>
    <li>one-one
        <ol>
        <li>one-one-one</li>

        <li>one-one-two</li>
        <li>one-one-three</li>
        </ol>
    </li>
    <li>one-two</li>
    <li>one-three</li>

    </ol>
</li>
<li>two</li>
<li>three
    <ol>
    <li>three-one</li>

    <li>three-two</li>
    <li>three-three</li>
    </ol>
</li>
</ol>
```

ol>li) используются для того, чтобы не обрабатывались элементы вложенных *ненумерованных* списков.

Свойство 'counter-increment' Оно определяет величину, на которую увеличивается содержимое счетчика при каждом новом вхождении элемента. По умолчанию приращение равно 1. Допускается использование отрицательных целых чисел.
Свойство 'counter-reset' Оно задает значение, которое присваивается счетчику при каждом новом вхождении элемента. По умолчанию оно равно 0.

```
<h1>This is a heading</h1>
<p>The ::before pseudo-element inserts content before
the content of an element.</p>
<h1>This is a heading</h1>
<p><b>Note:</b> IE8 supports the content property only
if a !DOCTYPE is specified.</p>
```

```
h1::before {
    content: url(smiley.gif);
}
```



Виды верстки

- Блочная верстка
- Табличная верстка
- Верстка flexbox

Табличная верстка

Создаётся с помощью обычной таблицы, таблица делится на колонки, а колонки на ячейки, в каждой ячейке можно расположить то, что вам требуется шапка, меню, контент, подвал и всё что должно быть по задумке на сайте.

```
<body>
<table width="100%" cellpadding="5" cellspacing="0">
 <tr> <td height="30" colspan="3" bgcolor="#F0FC0A"> Заголовок сайта </td></tr>
 <tr> <td bgcolor="#990033" valign="top"> Левая колонка </td> <td bgcolor="#999966" valign="top">
    Контент </td> <td bgcolor="#09EDD6" valign="top"> Правая колонка </td> </tr>
 <tr> <td height="30" colspan="3" bgcolor="#cccccc"> Подвал страницы </td> </tr>
<table>
```

Блочная верстка

```
<body>
 <div id="wrapper">
      <div id="header">
     </div>
     <div id="sidebarL"> </div>
      <div id="sidebarR"> </div>
     <div id="content"> </div>
     <div id="footer"> </div>
</div>
 </body>
```

Табличная верстка

+	-
сама по себе верстка простая + легко обеспечить одинаковый вид во всех браузерах + резиновая верстка, при изменении разрешения экрана, таблица формируется автоматически, растягивая и сжимая ячейки.	- много лишнего кода, большой вес страниц -не каждый дизайн возможно создать - медленная загрузка страниц

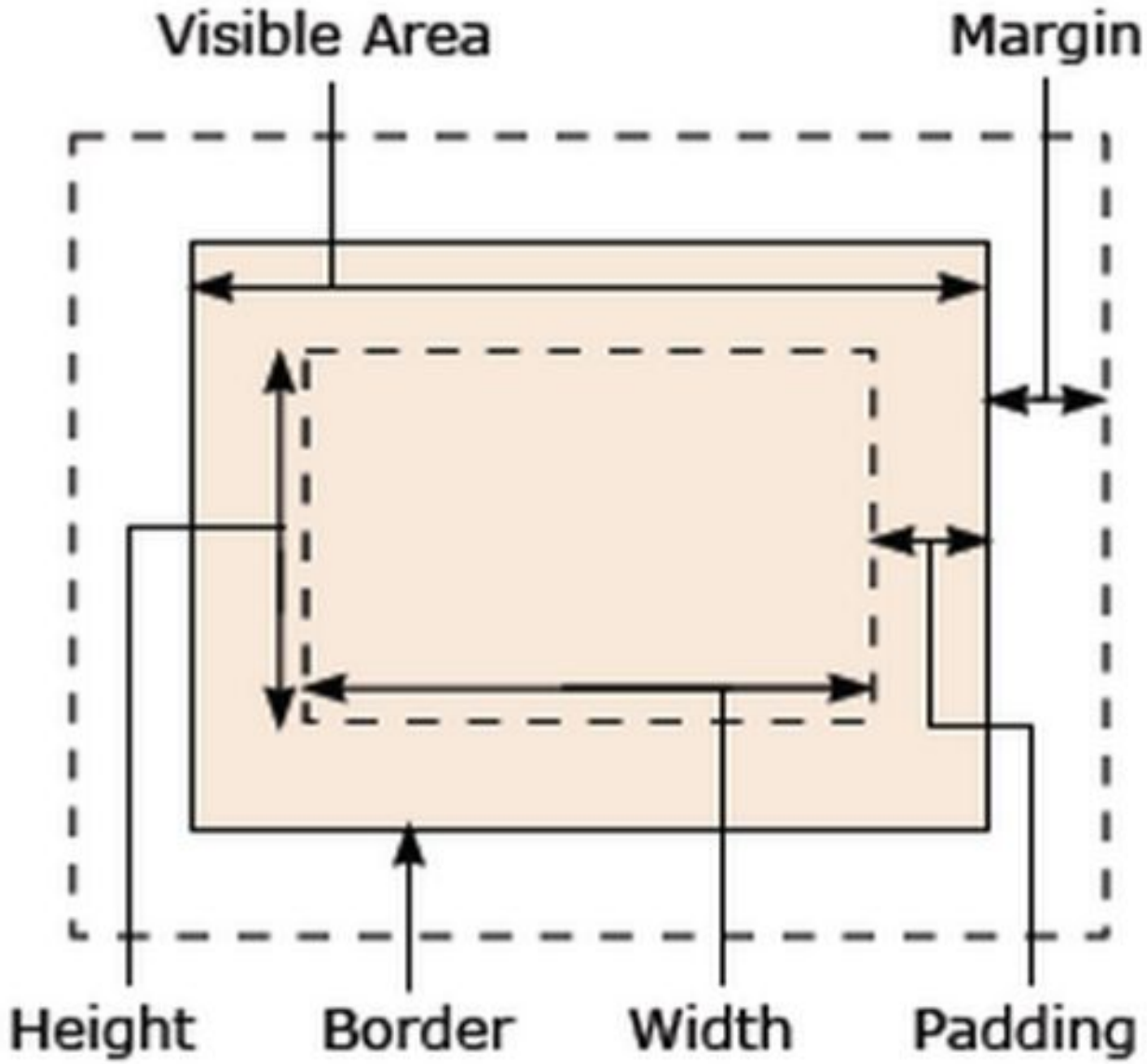
Блочная верстка

+	-
компактный код, небольшой вес страниц + отличная индексация поисковиками + слои можно накладывать друг на друга + быстрая загрузка страниц	сама по себе верстка сложная -трудно обеспечить одинаковый вид во всех браузерах -- при уменьшении разрешения экрана/браузера блоки съезжают друг на друга (нужна настройка в стилях под разрешения экрана)

Блочная верстка

 - является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида содержимого. Как правило, вид блока управляется с помощью стилей.

 - Обязателен!


Visible Area
Margin
Height
Border
Width
Padding

Display - Многоцелевое свойство, которое определяет, как элемент должен быть показан в документе.

block	Элемент показывается как блочный. Применение этого значения для встроенных элементов, например тега , заставляет его вести подобно блокам — происходит перенос строк в начале и в конце содержимого.
inline	Элемент отображается как встроенный. Использование блочных тегов, таких как и <p>, автоматически создает перенос и показывает содержимое этих тегов с новой строки. Значение inline отменяет эту особенность, поэтому содержимое блочных элементов начинается с того места, где окончился предыдущий элемент.
none	Временно удаляет элемент из документа. Занимаемое им место не резервируется и веб-страница формируется так, словно элемента и не было. Изменить значение и сделать вновь видимым элемент можно с помощью скриптов, обращаясь к свойствам через объектную модель. В этом случае происходит переформатирование данных на странице с учетом вновь добавленного элемента.
inline-block	Это значение генерирует блочный элемент, который обтекается другими элементами веб-страницы подобно встроенному элементу. Фактически такой элемент по своему действию похож на встраиваемые элементы (вроде тега <img>). При этом его внутренняя часть форматируется как блочный элемент, а сам элемент — как встроенный.
inline-table	Определяет, что элемент является таблицей как при использовании тега <table>, но при этом таблица является встроенным элементом и происходит ее обтекание другими элементами, например, текстом.
list-item	Элемент выводится как блочный и добавляется маркер списка.

Отступы

- **margin**

  Устанавливает величину отступа от каждого края элемента.

- **margin-top**

  Устанавливает величину отступа от верхнего края элемента.

- **margin-right**

  Устанавливает величину отступа от правого края элемента.

- **margin-left**

  Устанавливает величину отступа от левого края элемента.

- **margin-bottom**

  Устанавливает величину отступа от нижнего края элемента.

Поля

- **padding**

  Устанавливает значение полей вокруг содержимого элемента.

- **padding-top**

  Устанавливает значение поля от верхнего края содержимого элемента.

- **padding-right**

  Устанавливает значение поля от правого края содержимого элемента.

- **padding-left**

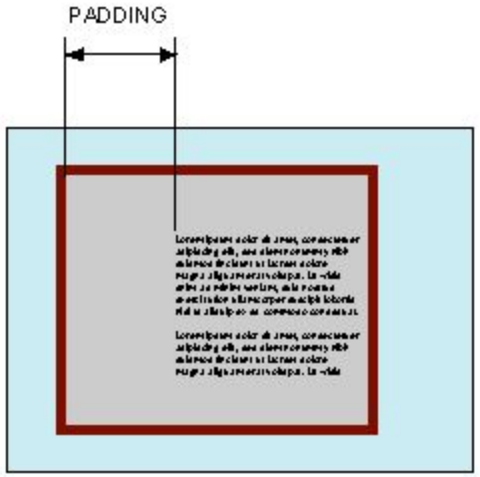
  Устанавливает значение поля от левого края содержимого элемента.

- **padding-bottom**

  Устанавливает значение поля от нижнего края содержимого элемента.

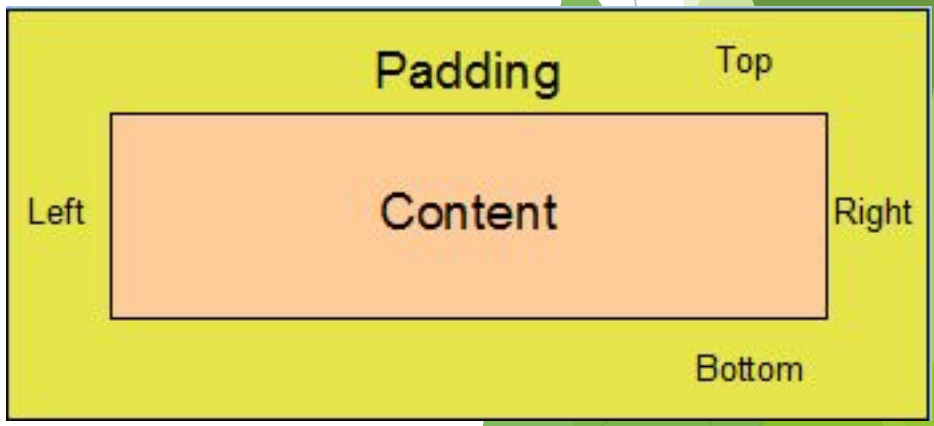
Padding

Устанавливает значение полей вокруг содержимого элемента. Полем называется расстояние от внутреннего края рамки элемента до воображаемого прямоугольника, ограничивающего его содержимое



Padding-left: 20px;

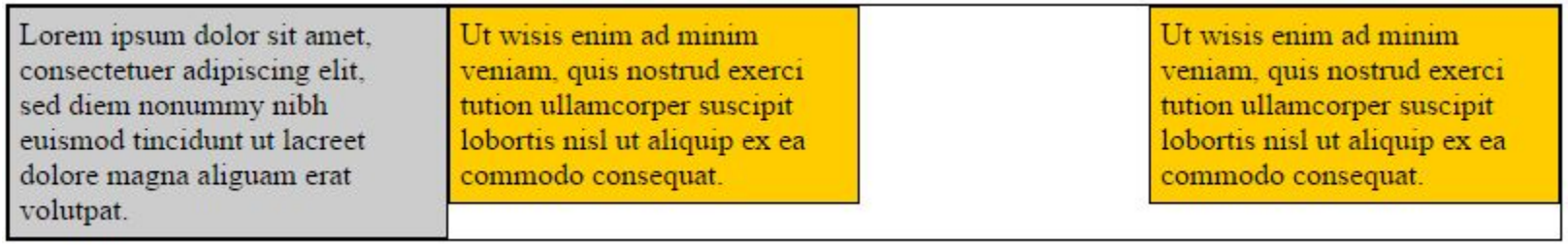
Padding-top – отступ сверху
Padding – right – отступ справа
Padding – bottom - отступ снизу
Padding – left – отступ слева



float

Left - Выравнивает элемент по левому краю, а все остальные элементы, вроде текста, обтекают его по правой стороне.

Right - Выравнивает элемент по правому краю, а все остальные элементы обтекают его по левой стороне.

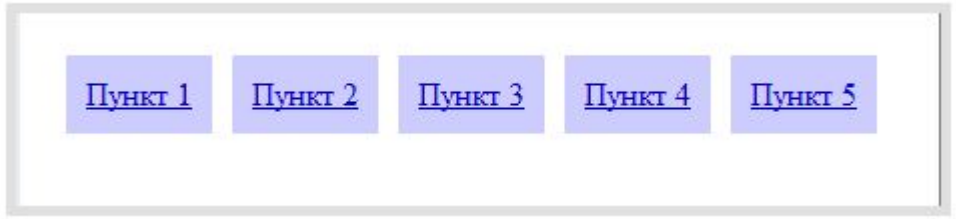


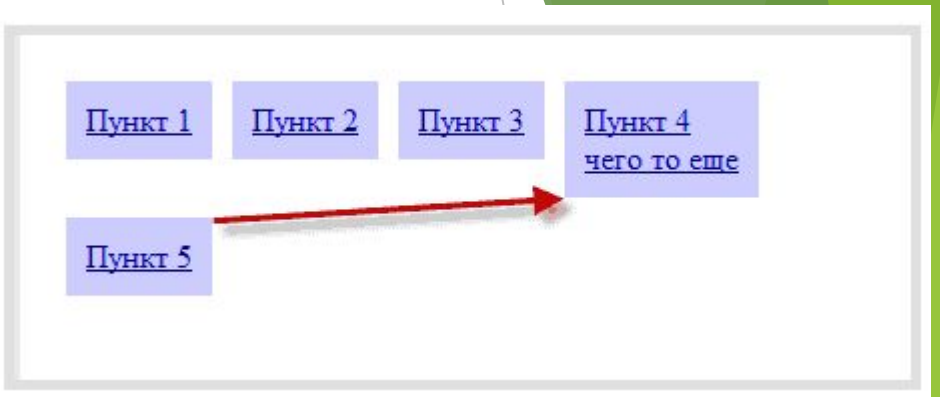
None -Обтекание элемента не задается. (по умолчанию)

Inherit - Наследует значение родителя.

Пример

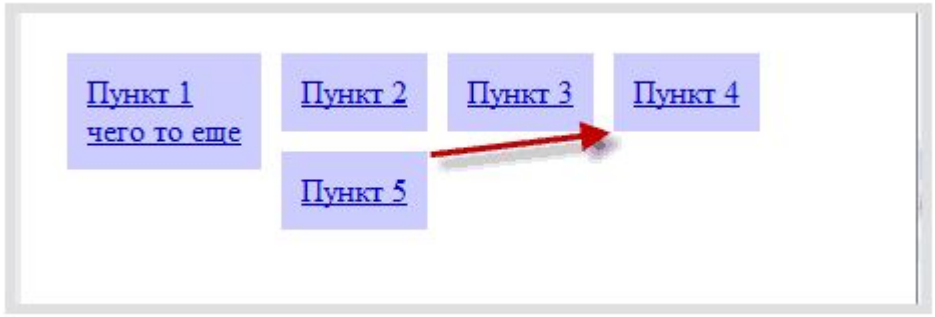
float:left;





нижняя граница предыдущего плавучего блока

А если высота первого пункта оказалась бы больше



Обтекание и очистка

```
<body>
<div> one</div><div> tow</div><div> three</div><div> fore</div>
<div class="clear"></div>
<div class="content">"
    <p> привязывается к указанной свойствами left, top,
        right и bottom точке на экране и не меняет своего
        положения при прокрутке веб-страницы
    </p>
</div>
</body>
```

```
<style>
    div{
        float: left;
        background: cornflowerblue;
        margin: 5px;
        width:200px;
    }
.clear{
    clear: both;
    width: 0px;
}
    .content{
        width:800px;
    }
</style>
```

one tow three fore

привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы

one tow three fore

привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы

position

- Absolute - абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и нет
- Положение элемента задается свойствами:

  left

  top

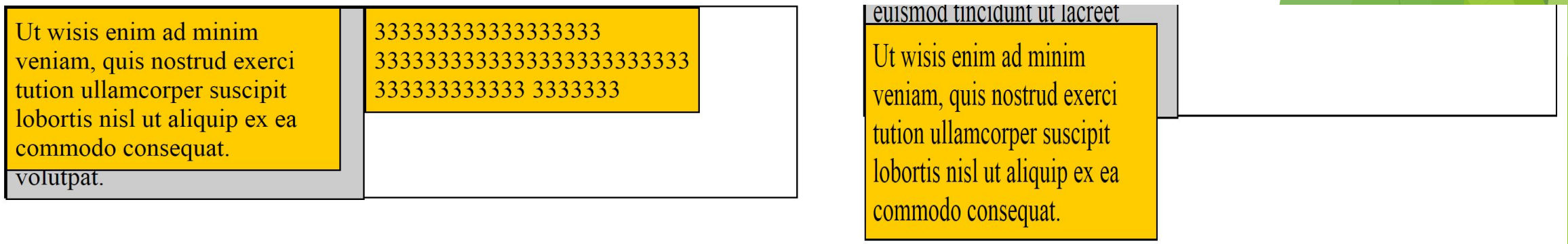
  right

  bottom

Так, если у родителя значение positionустановлено как static или родителя нет, то отсчет координат ведется от края окна браузера. Если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется от края родительского элемента.

fixed

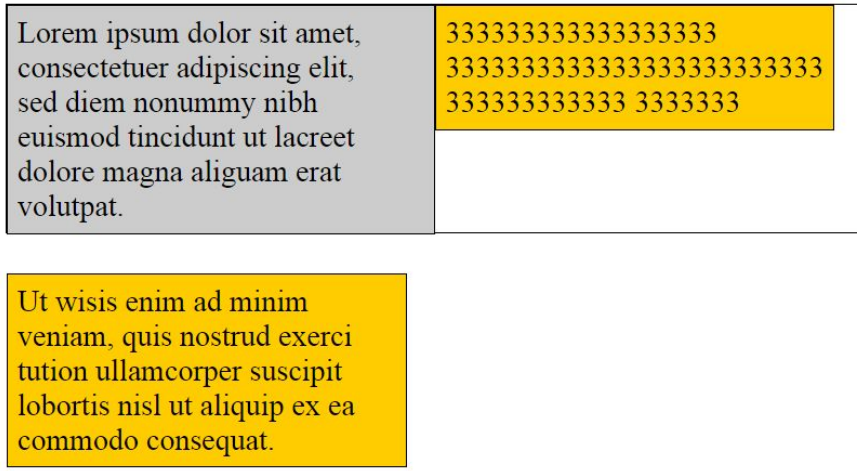
привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы



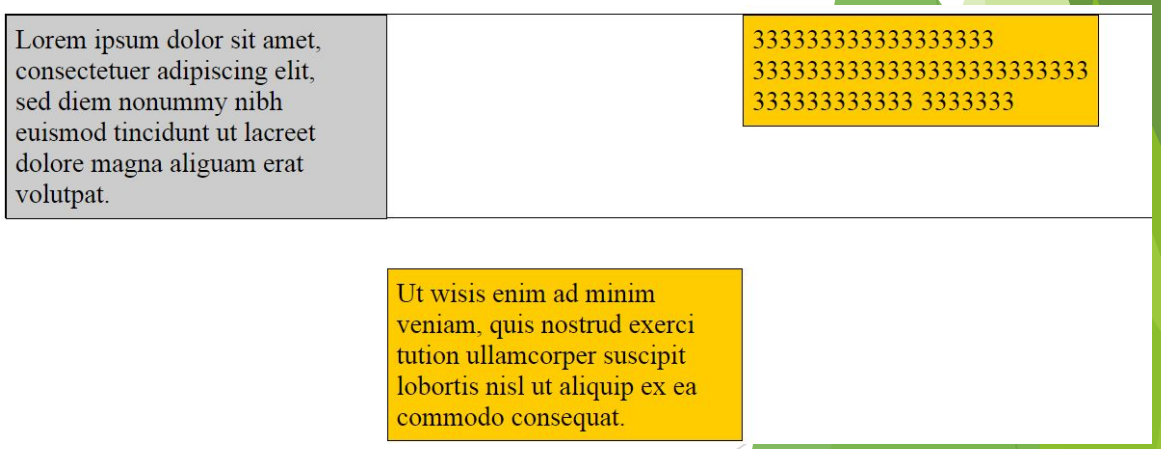
relative

Положение элемента устанавливается относительно его исходного места. Добавление свойств left, top, right и bottom изменяет позицию элемента и сдвигает его в ту или иную сторону от первоначального расположения.

Absolute



relative



static

Элементы отображаются как обычно. Использование свойств left, top, right и bottom не приводит к каким-либо результатам.

inherit

Наследует значение родителя.

overflow

Версии CSS

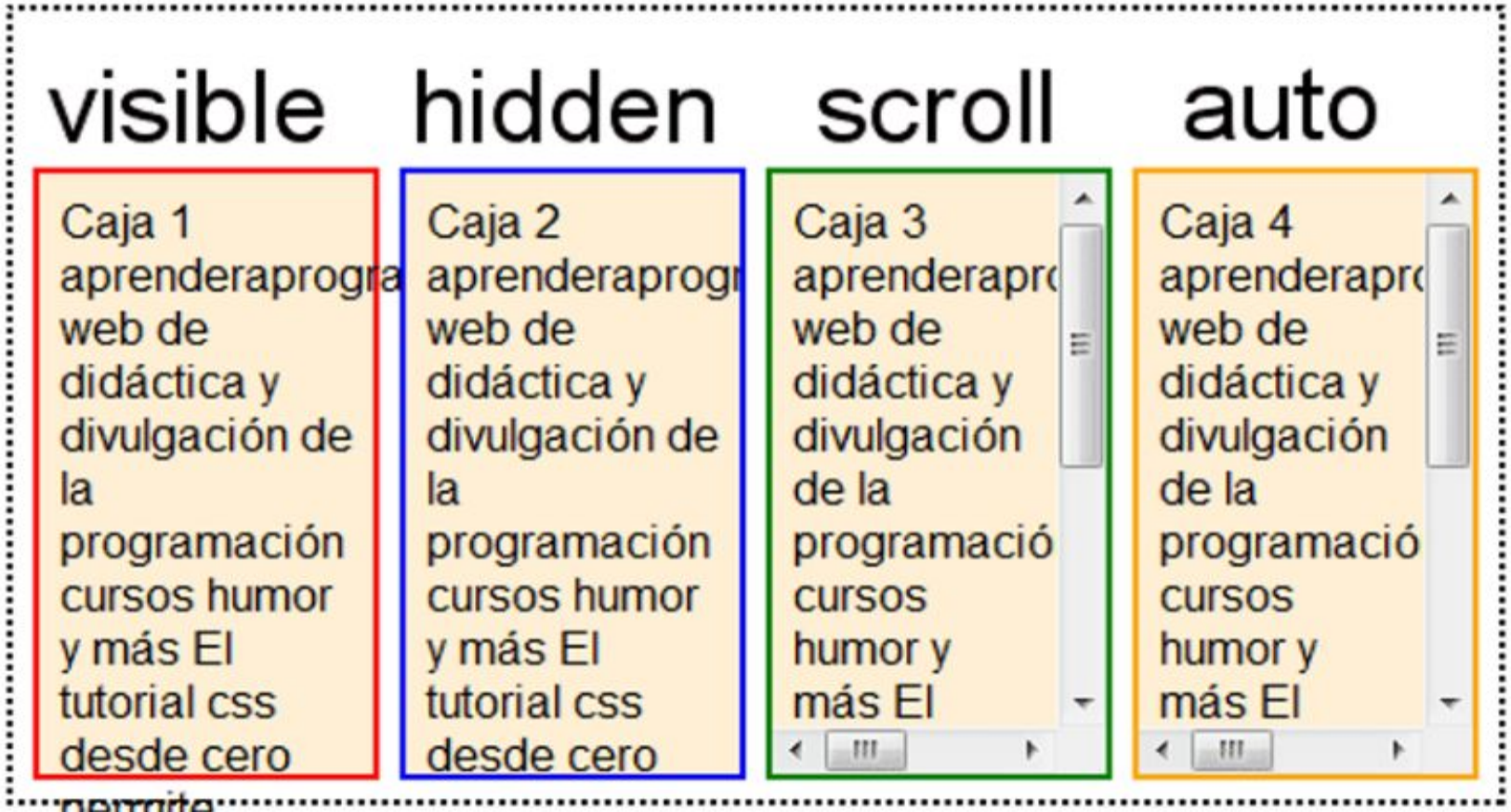
CSS 1	CSS 2	CSS 2.1	CSS 3
–	✔	✔	✔

Свойство overflow управляет отображением содержания блочного элемента, если оно целиком не помещается и выходит за область заданных размеров.

Значения

visible	Отображается все содержание элемента, даже за пределами установленной высоты и ширины.
hidden	Отображается только область внутри элемента, остальное будет скрыто.
scroll	Всегда добавляются полосы прокрутки.
auto	Полосы прокрутки добавляются только при необходимости.
inherit	Наследует значение родителя.

overflow css



z-index

Любые позиционированные элементы на веб-странице могут накладываться друг на друга в определенном порядке, имитируя тем самым третье измерение, перпендикулярное экрану. Каждый элемент может находиться как ниже, так и выше других объектов веб-страницы, их размещением по z-оси и управляет z-index.

Для position задано как absolute, fixed или relative.

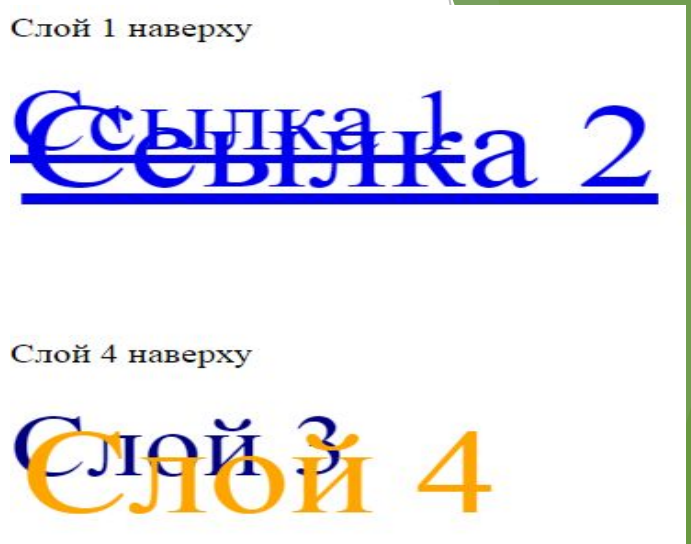
z-index: число | auto | inherit

Число - Чем больше значение, тем выше находится элемент по сравнению с теми элементами, у которых оно меньше.

auto — порядок элементов в этом случае строится автоматически, исходя из их положения в коде HTML

inherit указывает, что оно наследуется у родителя.

Пример



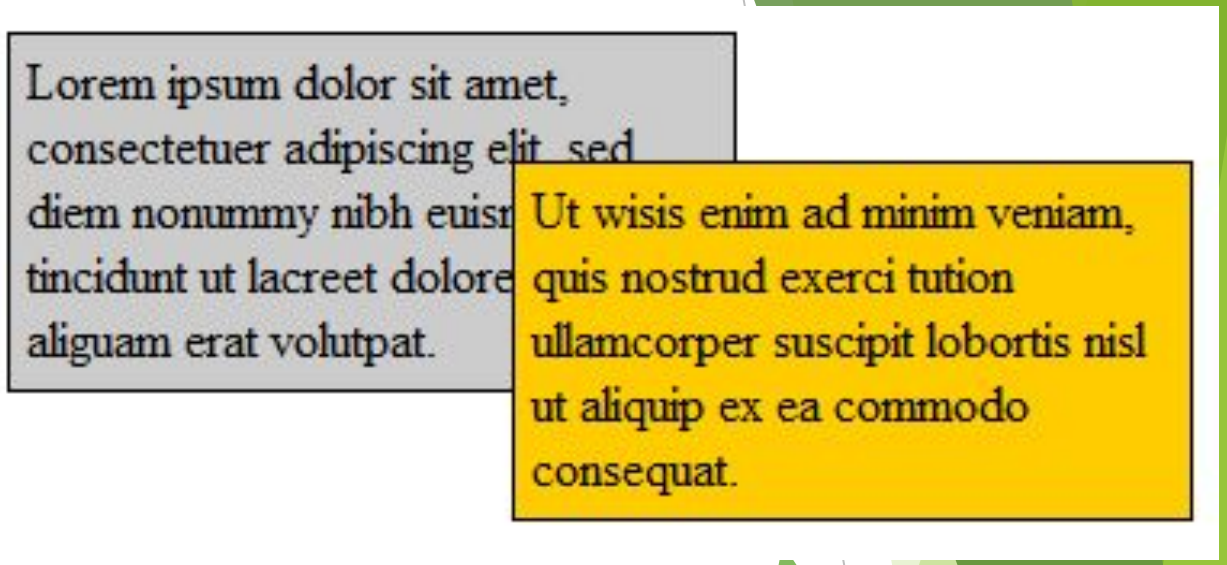

```
<p>Слой 1 наверху</p>
 <div id="layer1"><a href="mmm.php">Ссылка
   1</a></div>
 <div id="layer2"><a href="222.php">Ссылка
   2</a></div>
 <p>Слой 4 наверху</p>
 <div id="layer3">Слой 3</div>
 <div id="layer4">Слой 4</div>
```

```
#layer1 { z-index: 2; }
  #layer2 { z-index: 1; }
  #layer3 { z-index: 3; }
  #layer4 { z-index: 4; }
```

```
#layer1, #layer2, #layer3, #layer4 {
   position: relative; /* Относительное позиционирование */
```

Пример

```
<style type="text/css">
.block1 {
        width: 200px;
        background: #ccc;
        padding: 5px;
        padding-right: 20px;
        border: solid 1px black;
        float: left;
        }
 .block2 {
        width: 200px;
        background: #fc0;
        padding: 5px;
        border: solid 1px black;
         float: left;
        position: relative;
        top: 40px;
        left: -70px;
}
</style>
```




```
 <div class="block1">Lorem ipsum dolor sit amet,
consectetuer
   adipiscing elit, sed diem nonummy nibh euismod
tincidunt ut lacreet
   dolore magna aliguam erat volutpat.</div>
  <div class="block2">Ut wisis enim ad minim veniam, quis
nostrud
   exerci tution ullamcorper suscipit lobortis nisl ut aliquip
ex
   ea commodo consequat.</div>
```

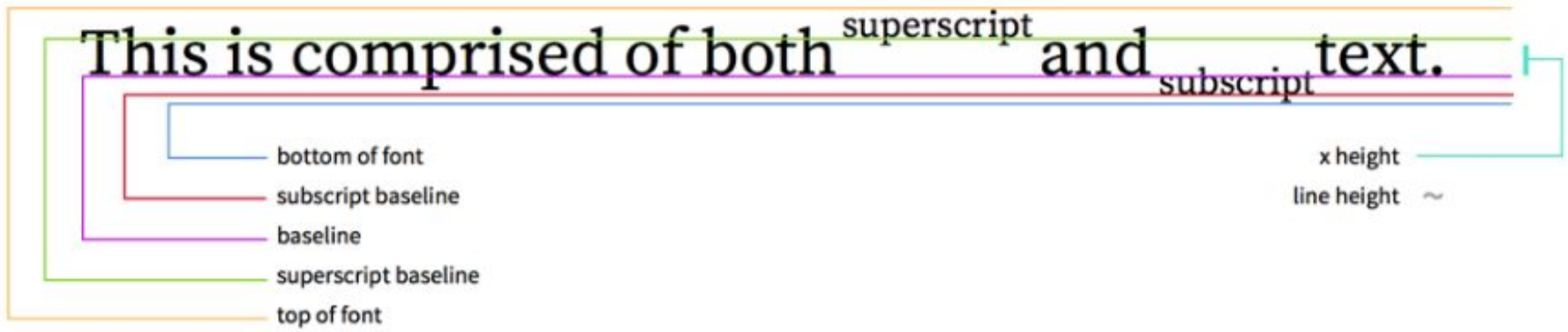
Шрифт

Свойство	Значение	Описание	Пример
font-family	имя шрифта	Задает список шрифтов	P {font-family: Arial, serif}
font-style	normal italic oblique	Нормальный шрифт Курсив Наклонный шрифт	P {font-style: italic}
font-variant	normal small-caps	Капитель (особые прописные буквы)	P {font-variant: small-caps}
font-weight	normal lighter bold bolder 100–900	Нормальная жирность Светлое начертание Полужирный Жирный 100 — светлый шрифт, 900 — самый жирный	P {font-weight: bold}
font-size	normal pt px %	нормальный размер пункты пикселы проценты	font-size: normal font-size: 12pt font-size: 12px font-size: 120%

Текст

Свойство	Значение	Описание	Пример
line-height	normal множитель значение %	Интерлиньяж (межстрочный интервал)	line-height: normal line-height: 1.5 line-height: 12px line-height: 120%
text-decoration	none underline overline line-through blink	Убрать все оформление Подчеркивание Линия над текстом Перечеркивание Мигание текста	text-decoration: none
text-transform	none capitalize uppercase lowercase	Убрать все эффекты Начинать С Прописных ВСЕ ПРОПИСНЫЕ все строчные	text-transform: capitalize
text-align	left right center justify	Выравнивание текста	text-align: justify
text-indent	значение %	Отступ первой строки	text-indent: 15px; text-indent: 10%

CSS-властивість	Значення	Пояснення
vertical-align:		**Для тексту вертикальне вирівнювання може приймати такі значення:**
	baseline;	по базові лінії;
	bottom;	по нижній лінії шрифта;
	middle;	по центру;
	sub;	нижній індекс;
	super;	верхній індекс;
	top;	по верхній лінії шрифта.



CSS-властивість	Значення	Пояснення
vertical-align:		**Для комірок таблиці:**
	baseline;	по базові лінії;
	bottom;	по нижній лінії шрифта;
	middle;	по центру;
	top;	по верхній лінії шрифта.

CSS-властивість	Значення	Пояснення
cursor:		**задає форму, яку прийматиме курсор, коли знаходитиметься над тегом:**
	url('шлях до курсору')	
	тип курсору;	

Вид	Значение	Тест	Пример
	default		P {cursor: default}
	crosshair		P {cursor: crosshair}
	help		P {cursor: help}
	move		P {cursor: move}
	pointer		P {cursor: pointer}

	progress		P {cursor: progress}
	text		P {cursor: text}
	wait		P {cursor: wait}
	n-resize		P {cursor: n-resize}
	ne-resize		P {cursor: ne-resize}
	e-resize		P {cursor :e-resize}
	se-resize		P {cursor: se-resize}
	s-resize		P {cursor: s-resize}
	sw-resize		P {cursor: sw-resize}
	w-resize		P {cursor: w-resize}
	nw-resize		P {cursor :nw-resize}

Цвет

Свойство	Значение	Описание	Пример
color	Цвет	Устанавливает цвет текста	P { color: #330000 }
background-color	Цвет transparent	Цвет фона	BODY { background-color: #6699FF }
background-image	URL none	Фоновый рисунок	BODY { background-image: url (bg.gif) }
background-repeat	repeat repeat-x repeat-y no-repeat	Повторяемость фонового рисунка	BODY { background-image: url (bg.gif) background-repeat: repeat-y }
background-attachment	scroll fixed	Прокручиваемость фона вместе с документом	BODY { background-image: url (bg.gif) background-attachment: fixed }
background-position	Проценты Пикселы top center bottom left right	Начальное положение фонового рисунка	BODY { background-position: left top }

2. Новые формы представления цвета

1) **RGBA** (Red Green Blue Alpha):
rgba(чер, зел, син, проз);
кольори задаються числами від 0 до 255,
прозорість – дробовим числом від 0.00 до 1.0

html:

```
<div class="one">One</div>
<div class="two">Two</div>
<div class="three">Three</div>
```

css:

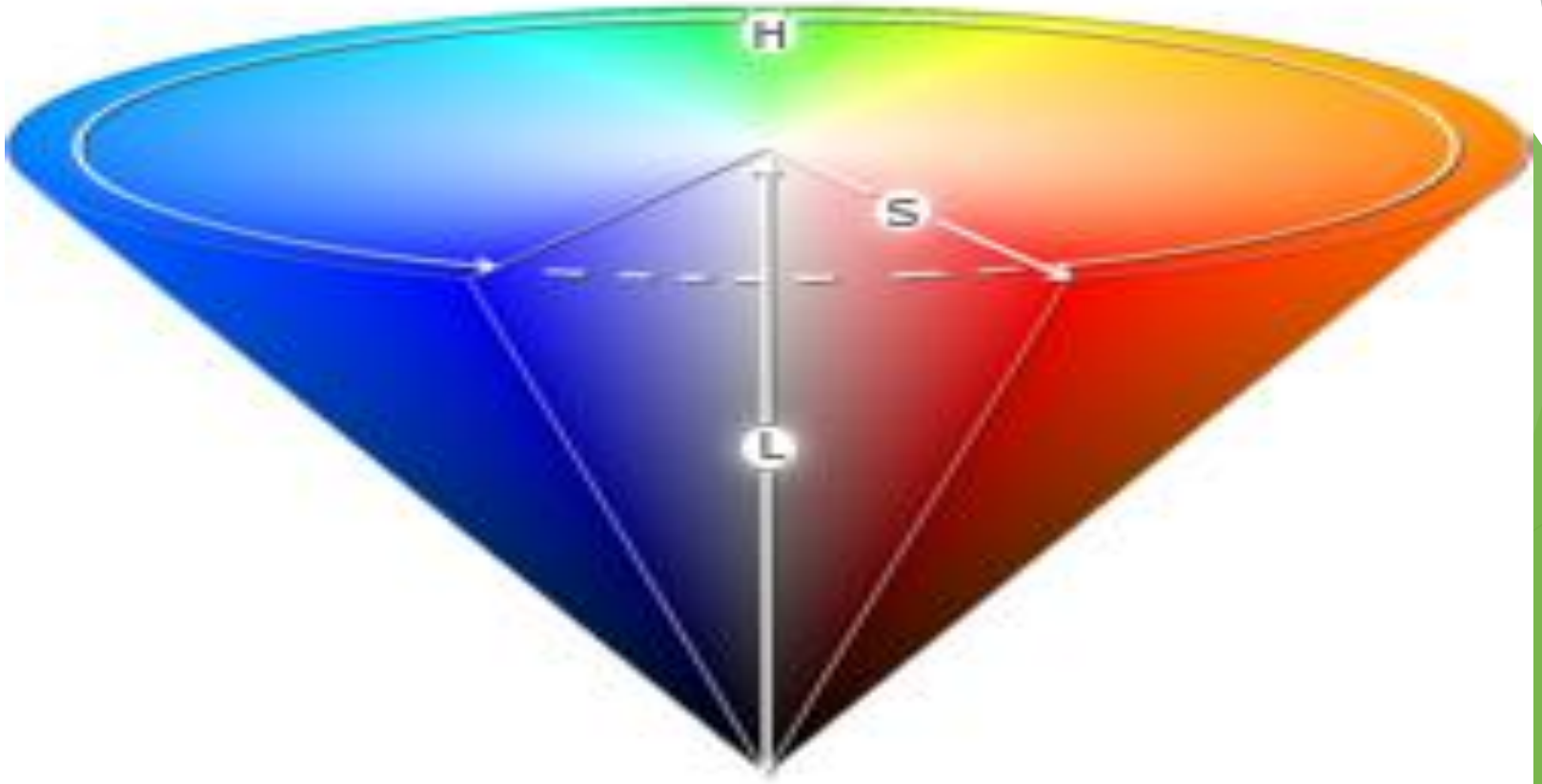
```
.one { left: 20px; top: 20px;
       background-color: rgba(255,0,0,0.5); }
.two { left: 50px; top: 50px;
       background-color: rgba(0,255,0,0.5); }
.three { left: 80px; top: 80px;
       background-color: rgba(0,0,255,0.5); }
.one, .two, .three {
    width: 100px; height: 100px;
    position: absolute; padding:5px; }
```

2) **HSL** (Hue, Saturation, Lightness):

hsl(тон, насыщенность, яркость);

тон задается числом от 0 до 360,

насыщенность и яркость– в процентах(от 0% до 100%)

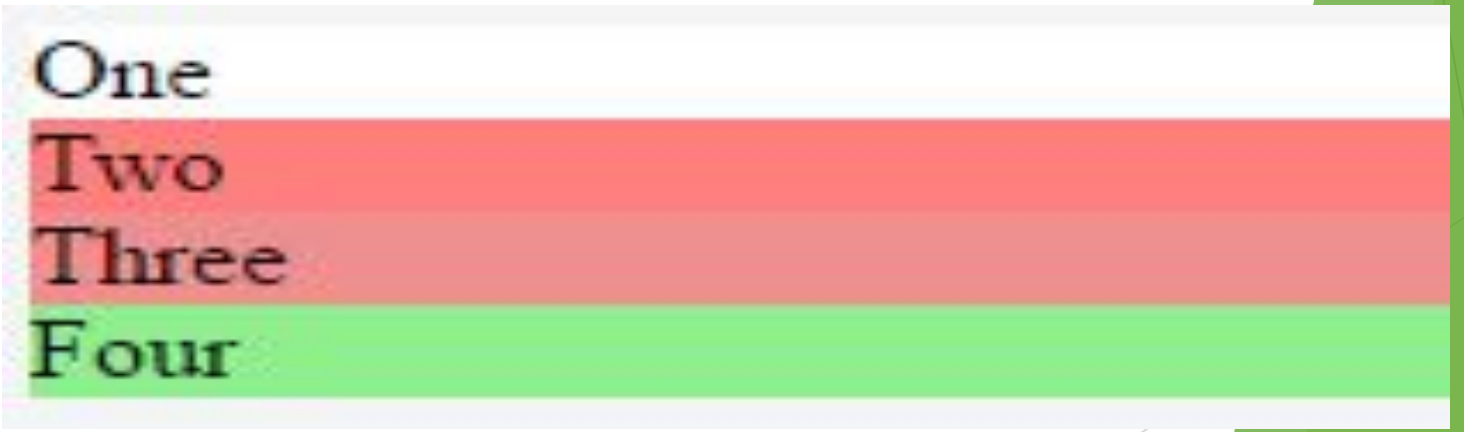


HTML:

```
<div class="a1">One</div>
<div class="a2">Two</div>
<div class="a3">Three</div>
<div class="a4">Four</div>
```

CSS:

```
.a1 { background-color: hsl(0, 100%, 100%);}
.a2 { background-color: hsl(0, 100%, 75%);}
.a3 { background-color: hsl(0,  75%, 75%);}
.a4 { background-color: hsl(120, 75%, 75%);}
```



3) **HSLA** (Hue, Saturation, Lightness, Alpha):

hsl(тон, насыщенность, яркость, прозрачность);

тон задается числом от 0 до 360,

Насыщенность и яркость – в процентах(от 0% до 100%),

прозрачность – дробным числом от 0.00 до 1.00

```
<p>HSLA colors:</p>
<p id="p1">Green</p>
<p id="p2">Light green</p>
<p id="p3">Dark green</p>
<p id="p4">Pastel green</p>
<p id="p5">Violet</p>
<p id="p6">Pastel violet</p>
```

```
#p1 {background-color:hsla(120,100%,50%,0.3);}
#p2 {background-color:hsla(120,100%,75%,0.3);}
#p3 {background-color:hsla(120,100%,25%,0.3);}
#p4 {background-color:hsla(120,60%,70%,0.3);}
#p5 {background-color:hsla(290,100%,50%,0.3);}
#p6 {background-color:hsla(290,60%,70%,0.3);}
```

HSLA colors:

Green

Light green

Dark green

Pastel green

Violet

Pastel violet

Списки

Свойство	Значение	Описание	Пример
list-style-type	disc circle square decimal lower-roman upper-roman lower-alpha upper-alpha none	Вид маркера. Первые три используются для создания маркированного списка, а остальные — для нумерованного.	LI {list-style-type: circle} LI {list-style-type: upper-alpha}
list-style-image	none URL	Устанавливает символом маркера любую картинку.	LI {list-style-image: url(check.gif)}
list-style-position	outside inside	Выбор положения маркера относительно блока строк текста.	LI {list-style-position: inside}
list-style		Универсальное свойство, включает одновременно все вышеперечисленные свойства.	

border

Универсальное свойство border позволяет одновременно установить толщину, стиль и цвет границы вокруг элемента

border: [border-widthborder: [border-width || border-styleborder: [border-width || border-style || border-color] |.

border-width определяет толщину границы.

Рис.1. Стили рамок

border-color устанавливает цвет границы

5. CSS свойства

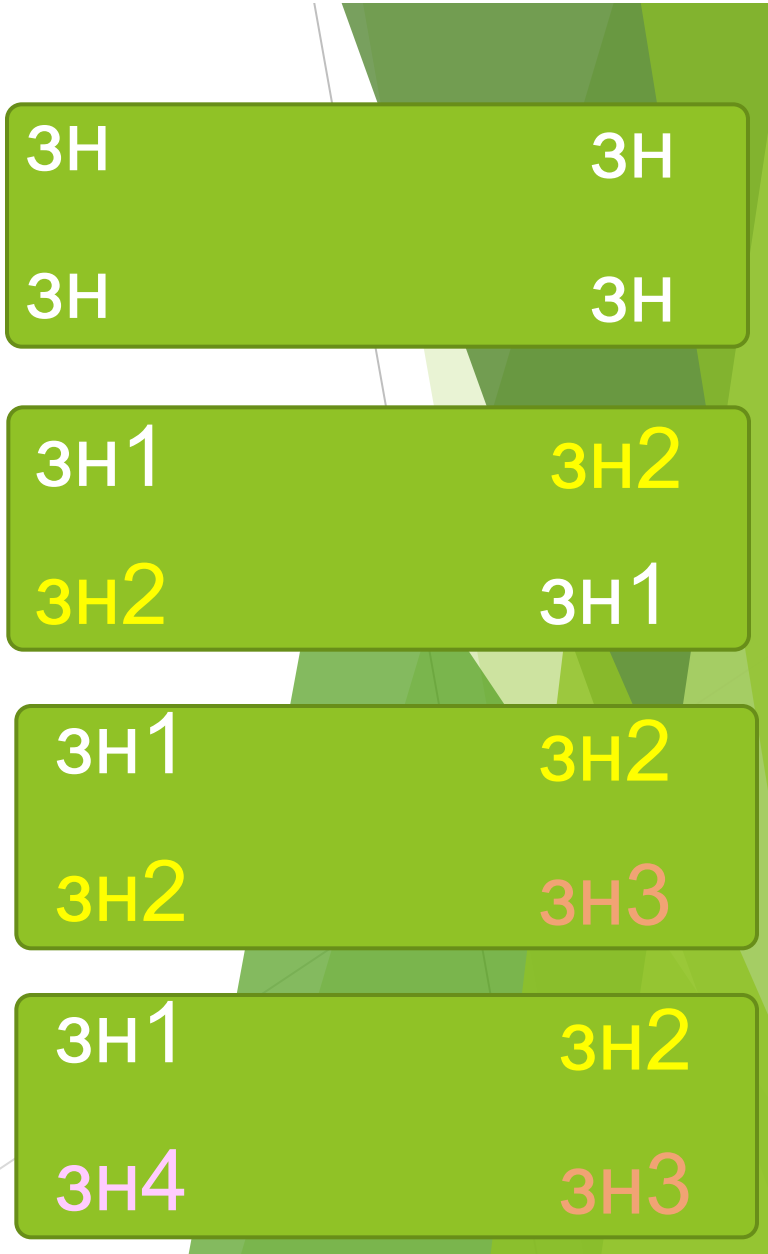
1) **Заокругленые концы тега**

border-radius: зн;

border-radius: зн1 зн2;

border-radius: зн1 зн2 зн3;

border-radius: зн1 зн2 зн3 зн4;



background: teal;
border-radius: 10px;

background: green;
border-radius: 10px 30px 60px;

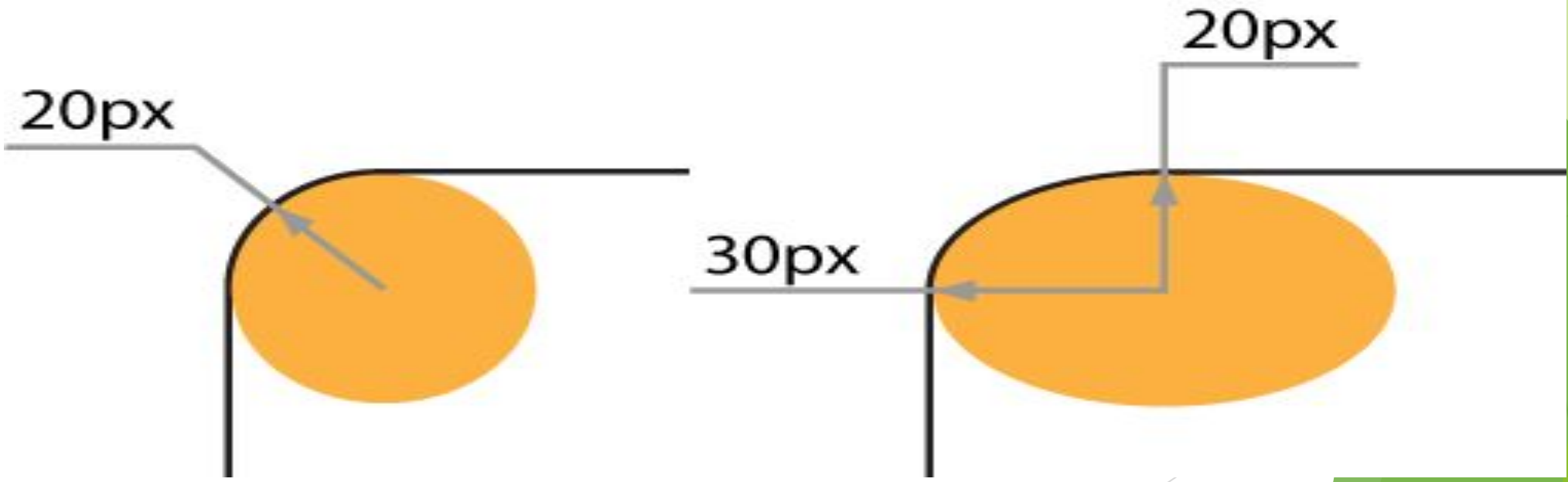
border-radius: 10px 40px;
background: orange;

border-radius: 10px 30px 60px 90px;
background: lightcoral;

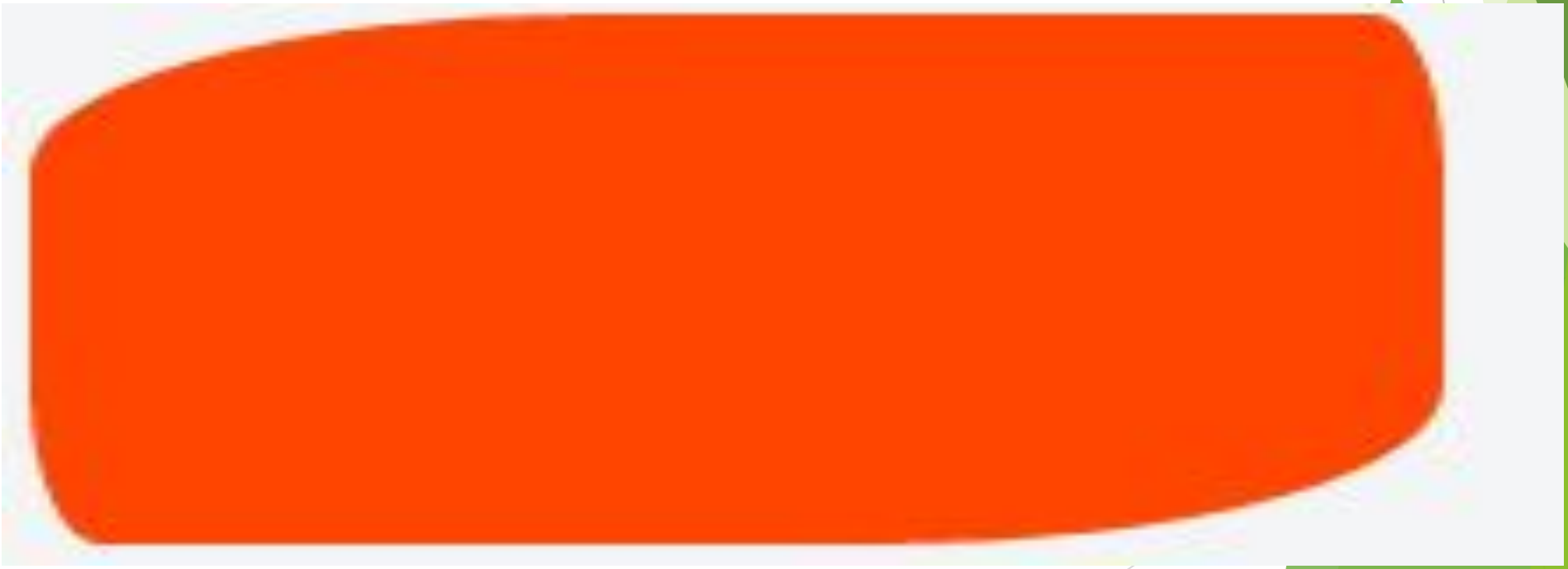
border-radius позволяет дополнительно через слеш задать от 1 до 4 значений.

```
border-radius: заокругленняX/заокругленняY;
```

```
border-radius: 30px/20px;
```



```
div{
    height: 100px;
    width: 200px;
    background: orangered;
    border-radius: 80px 10px/30px;
}
```



Значення **border-radius** можна задавати у відсотках (відсотки обчислюються від ширини тега)

```
div{
    height: 100px;
    width: 100px;
    background: green;
    border-radius: 50%;
}
```

```
div{
    height: 100px;
    width: 200px;
    background-image: url('1.jpg');
    border-radius: 50%;
    border: 3px solid blue;
}
```

2) Тень от **тега**

box-shadow: *змХ змУ розм розт кол;*

змХ – смещение по горизонтале;

змУ – смещение по вертикале;

розм – розмытие тени (чем меньше значение, тем четче тень);

розт – ростягивание тени:

0 – тень отвечает размерам тега;

<0 – тень меньше размеров тега;

>0 – тень больше размеров тега);

кол – цвет тени;

```
div{
     height: 100px;
     width: 100px;
     background: red;
     box-shadow: 20px 20px orange;
}
```

```
div{
     height: 100px;
     width: 100px;
     background: red;
     box-shadow:
               20px 20px 10px orange;
}
```

```css
div{
    height: 100px;
    width: 100px;
    background: red;
    box-shadow: 20px 20px 10px 10px orange;
}
```

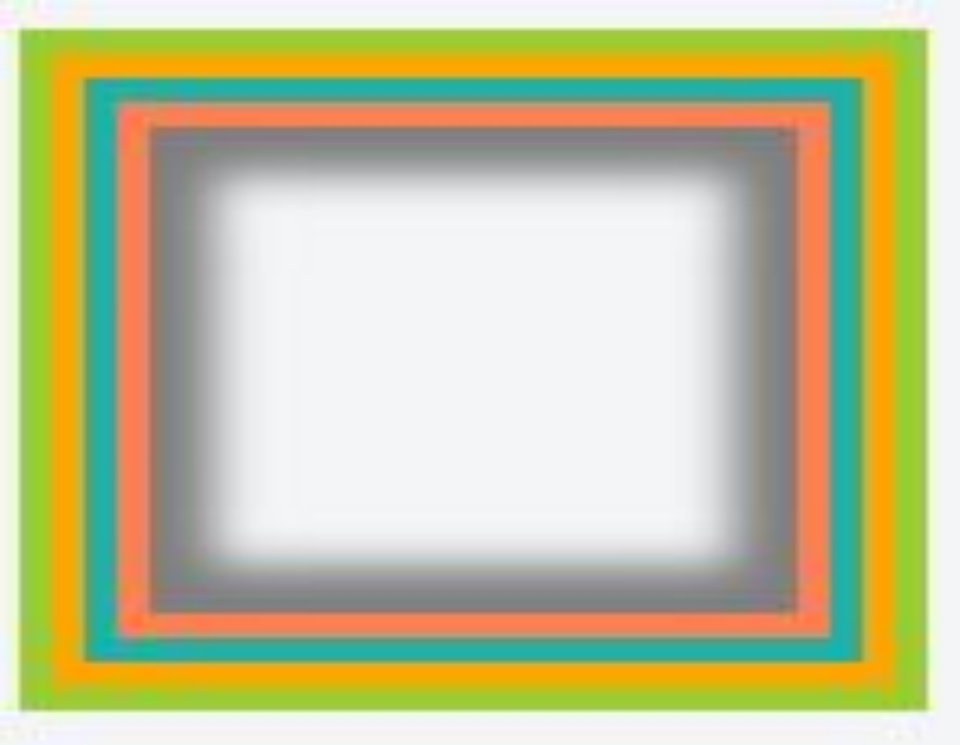
```css
div{
    height: 100px;
    width: 100px;
    background: red;
    box-shadow: 20px 20px 10px -10px orange;
}
```

```css
div{
    height: 100px;
    width: 100px;
    background: red;
    box-shadow: inset 0 0 10px 10px orange;
}
```

Тінь всередину тега

Можна задавати кілька тіней, розділяючи їх комою:

box-shadow: тінь1, тінь2, …;



```
div{
    margin:40px;
    height: 100px;
    width: 100px;
    box-shadow:
       0 0 0 5px coral,
       0 0 0 10px lightseagreen,
       0 0 0 15px orange,
       0 0 0 20px yellowgreen,
       inset 0 0 10px 10px gray;
}
```

3) Тінь від тексту

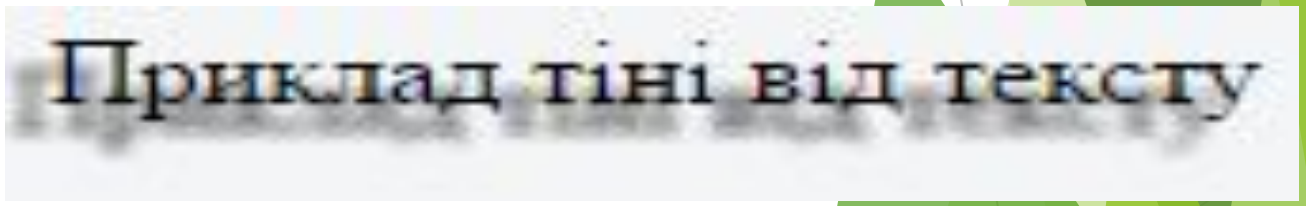
text-shadow: *змX змY розм кол;*

змX – зміщення по горизонталі;

змY – зміщення по вертикалі;

розм – розмиття тіні (чим менше значення, тим чіткіше тінь);

кол – колір тіні;




```
div{
    text-shadow: -5px 5px 3px rgba(0,0,0,0.75);
}
```

Можна задавати кілька тіней, вказуючи їх через кому: `text-shadow: тінь1, тінь2, …;`

Таблицы

- border-collapse - Устанавливает, как отображать границы вокруг ячеек таблицы.

collapse	Линия между ячейками отображается только одна, также игнорируется значение атрибута cellspacing.
separate	Вокруг каждой ячейки отображается своя собственная рамка, в местах соприкосновения ячеек показываются сразу две линии.
inherit	Наследует значение родителя.

- border-spacing
- Задает расстояние между границами ячеек в таблице. //border-spacing: 7px 11px;

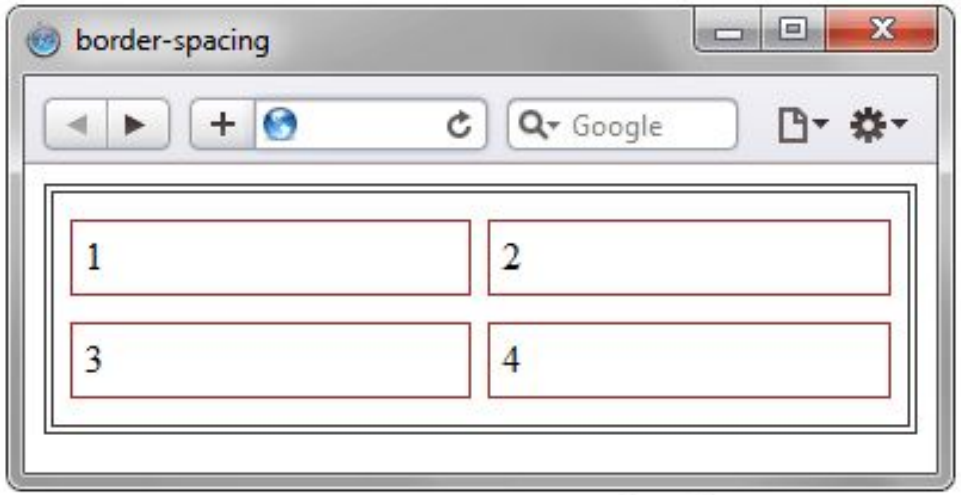


Рис. 1. Применение свойства border-spacing

- empty-cells

Задает отображение границ и фона в ячейке, если она пустая.

show	Отображает границу вокруг ячейки и фон в ней.
hide	Граница и фон в пустых ячейках не отображается. Если все ячейки в строке пустые, то строка прячется целиком.

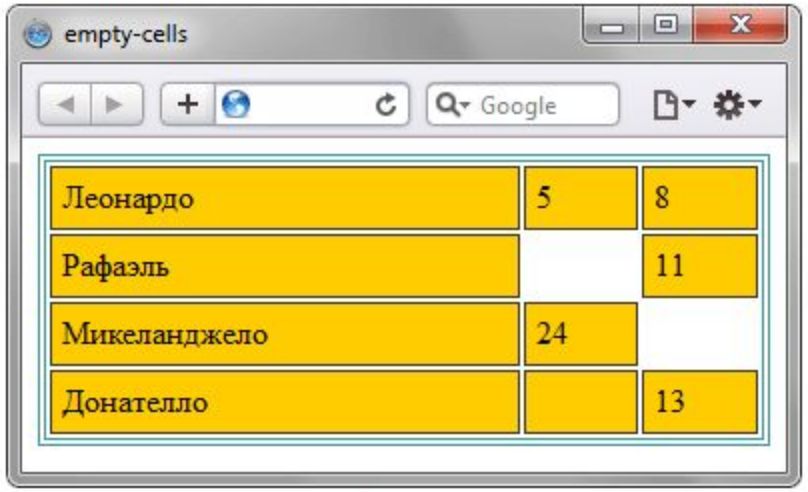


Рис. 1. Вид таблицы в браузере Safari

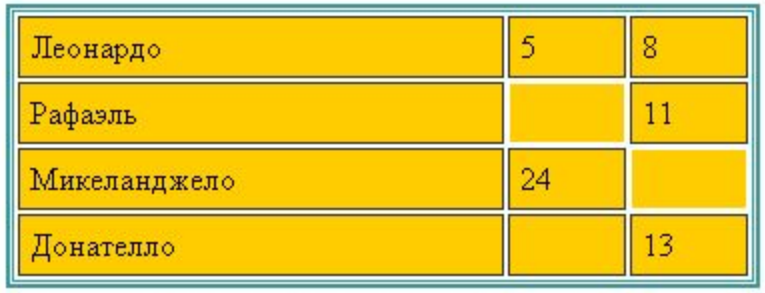


Рис. 2. Вид таблицы в браузере Internet Explorer 7

Ссылки

Псевдоклассы при работе со ссылками

Свойство	Описание
A:link	Определяет стиль для обычной непосещенной ссылки.
A:visited	Определяет стиль для посещенной ссылки.
A:active	Определяет стиль для активной ссылки. Активной ссылка становится при нажатии на нее.
A:hover	Определяет стиль для ссылки при наведении на нее мышью.

Размеры

height
Устанавливает высоту элементов.
max-height
Устанавливает максимальную высоту элемента.
max-width
Устанавливает максимальную ширину элемента.
min-height
Задает минимальную высоту элемента.
min-width
Устанавливает минимальную ширину элемента.
WWidth
Устанавливает ширину блочных или заменяемых элементов

Мнемоники

Специальные символы отображаются в текстах в виде специальных слов (entities). Вот примеры некоторых мнемоник:

< >	< >
™ ©	™ ©
π δ Δ	π δ Δ
И	&1048;
¢ £ ¥	¢ £ ¥
" &	" &

Полный список см.
http://www.w3schools.com/tags/ref_entities.asp

Приоритеты браузеров, которыми они руководствуются при обработке стилевых правил.

1. Стиль браузера.
2. Стиль автора.
3. Стиль пользователя.
4. Стиль автора с добавлением !important.
5. Стиль пользователя с добавлением !important.

- !important
- Ключевое слово !important играет роль в том случае, когда пользователи подключают свою собственную таблицу стилей.

Специфичность

Если к одному элементу одновременно применяются противоречивые стилевые правила, то более высокий приоритет имеет правило, у которого значение специфичности селектора больше.

За каждый идентификатор (в дальнейшем будем обозначать их количество через a) начисляется 100, за каждый класс и псевдокласс (b) начисляется 10, за каждый селектор тега и псевдоэлемент (c) начисляется 1. Складывая указанные значения в определённом порядке, получим значение специфичности для данного селектора.

```
*               {} /* a=0 b=0 c=0 -> специфичность = 0   */
li              {} /* a=0 b=0 c=1 -> специфичность = 1   */
li:first-line   {} /* a=0 b=0 c=2 -> специфичность = 2   */
ul li           {} /* a=0 b=0 c=2 -> специфичность = 2   */
ul ol+li        {} /* a=0 b=0 c=3 -> специфичность = 3   */
ul li.red       {} /* a=0 b=1 c=2 -> специфичность = 12  */
li.red.level    {} /* a=0 b=2 c=1 -> специфичность = 21  */
#t34            {} /* a=1 b=0 c=0 -> специфичность = 100 */
#content #wrap  {} /* a=2 b=0 c=0 -> специфичность = 200 */
```

Встроенный стиль, добавляемый к тегу через атрибут style, имеет специфичность 1000, поэтому всегда перекрывает связанные и глобальные стили. Однако добавление !important перекрывает в том числе и встроенные стили.

Селектор	Специфичность a-b-c-d	Правило №
*	0000	-
li	0001	4
li:first-line	0002	4
ul li	0002	4
ul ol+li	0003	4
table tr td.second	0013	3,4
h2.block.title.	0021	3,4
#xyz	0100	2
style=" "	1000	1

1. Самый высокий приоритет имеет атрибут **style**. Это правило перекрывает все селекторы описанные в стилях.
2. Второе место занимает присутствие **ID** в селекторе(#some-id).
3. Далее идут все атрибуты(в том числе и атрибут **class**) и псевдоклассы(pseudo-classes) в селекторе.
4. Самый низкий приоритет у селекторов с именами элементов и псевдоэлементами(pseudo-elements).

Задание

Каким цветом будут пункты списка и почему?

1. Какая специфичность будет у селектора table.forum tr:hover p?

1. 14
2. 22
3. 23
4. 32
5. 41

2. Какая специфичность будет у селектора #catalog .col3 .height div?

1. 301
2. 203
3. 121
4. 40
5. 31

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Список</title>
  <style>
   #menu ul li {
    color: green;
   }
   .two {
    color: red;
   }
  </style>
 </head>
 <body>
  <div id="menu">
   <ul>
    <li>Первый</li>
    <li class="two">Второй</li>
    <li>Третий</li>
   </ul>
  </div>
 </body>
</html>
```

Вопросы

1. В какой строке содержится ошибка?

1H1 { margin-left: 20px; }
2p { margin-left: 20px; padding-left: 20px; }
3h2 { margin-right: 20px; }
4head { color: #rob; }
5body { font-size: 11pt; color: #aaa; }

2. Таня для фона веб-страницы и цвета текста выбрала цвета #ffe9f2 и #6e143b и в стилях использовала следующий код, однако нужные цвета не проявились. В чем причина?

body {
background-color: #ffe9f2
color: #6e143b
}

1body написан строчными буквами.
2Свойство background-color неверное, следует писать background.
3Значения цветов указаны неправильно.
4В качестве селектора применять body некорректно.
5Не хватает точки с запятой.

3. Какая строка написана правильно?

1<P> { color: #333; }
2P { color: #333; }
3P: { color: #333;}
4P { color: 333; }
5P { color: #3333; }

Вопросы