Cascading Style Sheets

Спецификация CSS
http://www.w3.org/TR/CSS21/propidx.html

CSS

- ***CSS - Cascading Style Sheets*** - каскадные таблицы стилей
- 1994 г. – термин предложен Хокон Виум Ли
- позволяет отделить **представление** от **содержания** HTML-документа
- используется для одновременного управления стилями и компоновкой множества страниц Web
- CSS **не является** ни языком программирования, ни языком разметки, это - **система правил**, которые определяют, какие элементы HTML должны быть дополнительно оформлены.
- Консорциум W3C поддерживает стандарты каскадных таблиц стилей.

Стили

- впервые появились в HTML 4.0
- обычно хранятся в таблицах стилей

- Таблицы стилей могут быть определены
  - внутри HTML-документа,
  - в специальном файле с расширением ***.css***
- можно определить несколько стилей, которые, подчиняясь существующим правилам, будут каскадно задавать один определенный стиль.

Стили оформления:

- **стили шрифта** - для задания типа шрифта, размера и насыщенности;
- **стили текста** - для задания интервала между буквами и словами, высоты строк, горизонтального и вертикального интервала и абзацных отступов;
- **стили цвета** - для задания цвета фона и переднего плана;
- **стили рамок** - для вывода различных рамок, окружающих текстовые и графические элементы;
- **стили отступов** - для задания ширины различных отступов, окружающих текстовые и графические элементы;
- **стили фильтрации** - для применения специальных эффектов к текстовым и графическим элементам;
- **стили задания размера** - для задания высоты и ширины текстовых и графических контейнеров;
- **стили позиционирования** - для позиционирования элементов страницы в фиксированных пиксельных координатах на странице.

Преимущества использования CSS:

- Несколько дизайнов страницы для разных устройств просмотра.
- Уменьшение времени загрузки страниц сайта за счет переноса правил представления данных в отдельный CSS-файл.
- Простота последующего изменения дизайна.
- Дополнительные возможности оформления.

Недостатки

- Различное отображение верстки в различных браузерах (особенно устаревших.
- Необходимость на практике исправлять не только один CSS-файл, но и теги HTML и код PHP.

Комментарии CSS

```
/* текст комментария */
// однострочный комментарий
```

Комментарии могут охватывать несколько строк, и в этом случае браузер будет игнорировать эти строки.

Способы использования таблиц стилей

- **Линейная** (**in-line**) таблица стилей появляется внутри тега, к которому применяются ее объявления стилей; (внутри конкретного элемента HTML)
- **Встроенная** (**embedded**) таблица стилей является отдельным разделом стилей страницы Web, которая применяет свои стили ко всем определяемым на странице тегам; (внутри элемента `<head>` страницы HTML)
- **Внешняя** (**linked**) таблица стилей является внешним документом, содержащим задания стилей, которые применимы ко всем страницам, которые с ней соединены (во внешнем файле ***.css***).

Линейная таблица стилей

- когда особый стиль должен быть применен к единственному появлению элемента.
- Для описания используется атрибут `style` в соответствующем теге.
- Атрибут `style` может содержать любое свойство CSS.

```
<html>
…
<body>
…
   <p style="color: red; margin-left: 20px">
   Это параграф
   </p>
…
</body>
</html>
```

Встроенная таблица стилей

- когда один документ использует единый стиль.
- определяются в разделе заголовка `<head>` с помощью тега `<style>`.

```
<html>
<head>
<style type="text/css">
  body {background-color: red}
  p {margin-left: 20px}
</style>
</head>
…
</html>
```

Внешняя таблица стилей

- когда стиль применяется к нескольким страницам.
- позволяет изменить внешний вид всего Web-сайта, изменяя один файл.
- Каждая страница должна соединяться с таблицей стилей с помощью тега `<link>` (в разделе заголовка `<head>`)

```
<html>
<head>
<link rel="stylesheet" type="text/css"
  href="mystyle.css">
<link rel="stylesheet" type="text/css"
  href="http://www.htmlbook.ru/main.css">
</head>
…
</html>
```

Внешняя таблица стилей
(альтернативный вариант подключения)

- таблица стилей может быть подключена к веб-документу посредством директивы `@import`, располагающейся в этом документе между тегами `<style>` (в заголовке `<head>`)
- Все правила этой таблицы действуют на протяжении всего документа;

```
<head>
.....
  <style type="text/css" media="all">
  @import url(style.css);
  </style>
</head>
```

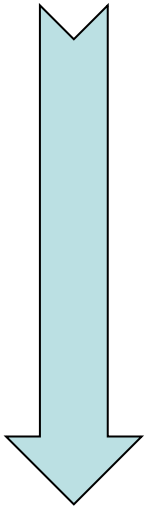
Примечания:

- внешнюю таблицу стилей можно создать в любом текстовом редакторе;
- файл с внешней таблицей стилей не должен содержать никаких тегов HTML;
- файл с внешней таблицей стилей необходимо сохранить с расширением *.css*.

Приоритет использования стилей

Если для элемента HTML определено более одного стиля, то все стили будут последовательно "каскадированы" в новую "виртуальную" таблицу стилей

- стили, используемые по умолчанию браузером; *(низкий)*
- стили, хранящиеся во внешней таблице;
- стили, хранящиеся во внутренней таблице стилей (внутри тега <head>);
- встроенный стиль (внутри элемента HTML). *(высокий)*



Синтаксис правил стилей

`селектор {свойство:значение;}`

- `Селектор` — элемент/тег HTML, который необходимо определить.
- `Свойство` — атрибут, значение которого задаем.
- Каждое свойство может принимать `значение`.

```
p {
  margin: 10px;
  font-family: “Times New Roman”;
  color: green;
}
```

- CSS не чувствителен к регистру, переносу строк, пробелам и символам табуляции, поэтому форма записи зависит от желания разработчика.
- **Замечание.** Имена селекторов обязательно должны начинаться с латинского символа (a-z, A-Z) и могут содержать в себе цифры.

- Свойство и значение разделяются двоеточием и помещаются внутри фигурных скобок:

```
p {font-size:75%;}
```

- Если значение состоит из нескольких слов, то необходимо поместить значение в кавычки:

```
h1 {font-family: "lucida calligraphy"}
```

- Если требуется определить более одного свойства, то необходимо разделить свойства точкой с запятой:

```
table {
    font-family: arial, "sans serif";
    border-style: dotted
}
```

- Чтобы определения стилей было удобно читать, можно каждое свойство писать на отдельной строке:

```
h2{
    font-family: arial;
    margin-right: 20pt;
    color:#ffffff
}
```

- При определении правил допускается группировка селекторов, при этом в качестве разделителя селекторов используется запятая. В примере в группу были объединены все элементы абзацев, таблиц и списков. Все эти элементы будут выведены шрифтом sans serif:

```
p,table,li {font-family: sans-serif;}
```

Виды селекторов

- **селектор элемента;**
```
p {font-family: Garamond, serif;}
```
- **селектор объединения в группу**
```
h2, p {color: red}
```
- **селектор класса;**
```
.note {color:red; background:yellow; font-weight: bold;}
```
- **селектор идентификатора;**
```
#paragraph1 {margin: 0;}
```
- **селектор атрибута;**
```
a[href="http://www.romantiki.ru"]{font-weight:bold;}
```
- **контекстный селектор;**
```
div#paragraph1 p.note {color: red;}
```
- **дочерний селектор;**
```
p.note > b {color: green;}
```
- **соседний селектор;**
```
h1 + p {font-size: 24pt;}
```
- **селектор псевдокласса;**
```
a:active {color:yellow;}
```
- **селектор псевдоэлемента.**
```
p:first-letter {font-size: 32pc;}
```
- ***универсальный селектор***, обозначающий любой элемент, встречающийся в документе. Перед любым селектором, задающим класс или идентификатор, можно поставить знак универсального селектора
```
* {color:red;}
```

Селектор элемента (тега)

- В качестве селектора может выступать любой тег HTML, для которого определяются правила форматирования (цвет, фон, размер и т.д.)

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

- соответствует всем элементам на странице с указанным названием (элементам `p`, в приведенном случае)

```
p {
    font-size: 110%;
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

Селектор объединения в группу

- применить одинаковое оформление к `h2` и `p`, тогда можно было бы написать следующий CSS:

```
h2 {color: red}
p {color: red}
```

- можно сократить код CSS, группируя селекторы вместе с помощью запятой - правила в скобках применяются к обоим селекторам:

```
h2, p {color: red}
```

Селектор класса

- соответствует всем элементам, которые имеют атрибут *class* с указанным значением.

```
тег.имя_класса {свойство1: значение; свойство2: значение; ...}
```

- Чтобы указать в коде HTML, что тег используется с определенным стилем, к тегу добавляется атрибут

```
class="имя_класса"
```

- Можно использовать классы и без указания тега.

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

- При такой записи, класс можно применять к любому тегу.

Пример

```
<style type="text/css">
    p {font-size: 110%;
        font-family: Verdana, Arial, Helvetica, sans-serif;
        }
    p.Color {color: blue;}
    .left {margin-left: 40pt;}
  </style>
</head>
<body>
   <p>Пример использования селектора тегов.</p>
   <p class="Color">Пример использования класса.</p>
   <table class="left" border="1">
            <tr><td>Эта таблица будет иметь
            <span class="Color"> внешний отступ</span>,
             равный 40 пунктам. </td></tr>
    </table>
    <p class="left">Этот параграф будет иметь внешний отступ,
            равный 40 пунктам.</p>
```

EXAMPLES\css_01.html

Селектор ID (Идентификатор)

- соответствует элементу, который имеет атрибут ***id*** с указанным значением
- определяет уникальное имя элемента, которое используется для изменения его стиля и обращения к нему через скрипты, что позволяет управлять стилем элемента динамически.

```
#Имя_идентификатора { свойство1: значение;
          свойство2: значение; ... }
```

- В отличие от классов, идентификаторы должны быть **уникальны** (встречаться в коде документа только один раз).
- Обращение к идентификатору происходит аналогично классам, но в качестве ключевого слова у тега используется параметр **`id`**, значением которого выступает имя идентификатора. Символ решетки при этом уже не указывается.

- Идентификаторы можно применять к конкретному тегу.

```
Тег#Имя идентификатора { свойство1: значение;
          свойство2: значение; ... }
```

Селектор ID (Идентификатор)

```
…
<head>
…
  <style type="text/css">
   #help {
    position: absolute;
    left: 160px;
    top: 50px;
        width: 225px;
    height: 180px;
    background: #f0f0f0;
        }
  </style>
</head>

<body>
<div id="help">Пример использования Идентификаторов </div>

</body>
…
```

EXAMPLES\css_02.html

Универсальные селекторы

- можно использовать для выбора каждого элемента на странице для применения к ним стилей оформления.

```
* { описание_правил_стиля }
```

для каждого элемента на странице должна быть добавлена сплошная красная граница толщиной 1 пиксель:

```
*{border: 1px solid #FF0000;}
```

EXAMPLES\css_04.html

Селекторы атрибутов

- **позволяют установить стиль тега по присутствию определенного параметра или его значения**
- поддерживаются браузером Internet Explorer начиная с версии 7.0, для корректной работы необходимо добавить правильный <!DOCTYPE>.

```
[атрибут] { описание_правил_стиля }
селектор[атрибут] { описание_правил_стиля }
```

В первом случае стиль применяется ко всем элементам, которые содержат указанный атрибут.
во втором – только к определенным селекторам.

создать красную границу вокруг любого изображения (`img`), которое имеет атрибут `alt`.

```
img[alt] {border: 1px solid #FF0000;}
```

EXAMPLES\css_08.html

- **Можно выбирать элементы также и по значению атрибута, а не только по названию атрибута.**

```
[атрибут="значение"] { описание_правил_стиля }
селектор[атрибут="значение"] { описание_правил_стиля }
```

создать черную границу вокруг изображения с атрибутом `src` со значением *alert.gif*:

```
img[src="alert.gif"] {border: 1px solid #000000;}
```

Дочерние селекторы
(Селекторы потомков)

- для выбора только определенных элементов, которые являются потомками других определенных элементов (в дереве элементов находится прямо внутри родительского элемента).
- не поддерживаются в браузере IE 6 (и более младших версиях).

```
Селектор_1 > Селектор_2 { Описание_правил_стиля }
```

- Стиль применяется к `Селектору_2`, но только в том случае, если он является дочерним для `Селектора_1`.
- По своей логике дочерние селекторы похожи на контекстные селекторы. Разница между ними следующая: Стиль к дочернему селектору применяется только в том случае, когда он является прямым потомком, иными словами, непосредственно располагается внутри родительского элемента. Для контекстного селектора допустим любой уровень вложенности.

правило задает цвет текста `green`, только тех `strong` элементов, которые являются потомками элементов `p`, но не для других элементов `strong`:

```
p > strong {color: green;}
```

EXAMPLES\css_111.html

Контекстные селекторы

- селекторы, которые работают только в определенном контексте.
- Контекстный селектор состоит из простых селекторов разделенных пробелом.

```
тег1 тег2 { ... }
```

- В этом случае стиль будет применяться к `тегу2`, когда он размещается внутри `тег1`.
- Не обязательно контекстные селекторы содержат только один вложенный тег. В зависимости от ситуации допустимо применять два и более последовательно вложенных друг в друга тегов.
- Более широкие возможности контекстные селекторы дают при использовании идентификаторов и классов. Это позволяет устанавливать стиль только для того элемента, который располагается внутри определенного класса.

Контекстные селекторы

- определенные элементы, которые являются нижележащими относительно других конкретных элементов (не просто прямыми потомками, но также расположенные ниже в дереве)
- очень похожи на селекторы потомков, за исключением того, что селекторы потомков выбирают только непосредственно нижележащих, а контекстные селекторы выбирают подходящие элементы в любом месте иерархии элементов, а не только непосредственно нижележащих.

```
<div>
  <em>Привет</em>
  <p>и сразу же
     <em>Пока</em>.
  </p>
</div>
```

- элемент `div` является предком всех других элементов. Он имеет двух потомоков, `em` и `p`. Элемент `p` имеет один элемент-потомок `em`.

EXAMPLES\css_000.html

Контекстные селекторы

```
… <head> …
<style type="text/css">
   p b {
     font-weight: bold; color: red;
   }
 </style>
</head>
<body>
   <div>
    <b>Жирное начертание текста</b>
   </div>
   <p>
    <b>Одновременно жирное начертание текста и
          выделенное цветом
    </b>
   </p>
</body>
…
```

В данном примере показано обычное применение тега <b> и этого же тега, когда он вложен внутрь параграфа <p>. При этом меняется цвет текста

Соседние селекторы

(Селекторы смежных одноуровневых элементов)

- позволяют выбирать определенный элемент, который следует в коде документа непосредственно после другого определенного элемента.
- Не поддерживаются в браузере IE 6 (и более младших версиях).

```
<p>Это <b>пример</b> <var>соседних</var>
                селекторов.</p>
```

Теги `<var>` и `<b>` представляют собой соседние элементы.

```
селектор_1 + селектор_2 { описание_правил_стиля }
```

- стиль применяется к `селектору_2`, но только в том случае, если он является соседним для `селектора_1` и следует сразу после него.
- выбрать элементы `p`, которые следуют непосредственно после элементов `h2`, но никакие другие элементы `p`:

```
h2 + p { ...}
```

Псевдоклассы

- используются для обеспечения стилевого оформления не для элементов, а для различных состояний элементов.

```
элемент:псевдокласс { описание_правил_стиля }
```

- Наиболее обычным применением, которое можно встретить, является оформление состояний ссылок.

  Различные псевдо-классы и описание состояния ссылки, которое они выбирают:

`:link` - обычное состояние непосещенных ссылок по умолчанию

`:visited` – посещенные ссылки

`:hover` - ссылка, на которой в данный момент находится указатель мыши

`:active` - ссылка, на которой в данный момент происходит щелчок

`:focus` - ссылка (или поле формы, или что-то еще), в которой в данный момент находится курсор (клавиатуры)

Псевдоклассы

Следующие правила CSS определяют что:

- по умолчанию ссылки будут синими.
- когда курсор мыши оказывается над ссылкой, используемое по умолчанию подчеркивание ссылки исчезает.
- когда ссылка будет посещена, она станет серой.
- когда ссылка активна, она становится жирной, как дополнительный признак, что что-то сейчас произойдет.

```
a:link {color: blue;}
a:visited {color: gray;}
a:hover {text-decoration: none;}
a:active {font-weight: bold;}
```

Обратите внимание, если вы не определите эти правила в том же порядке, как они показаны выше, они могут работать не так, как вы ожидаете. Это обусловлено правилом специфичности, заставляющем более поздние правила в таблице стилей переопределять более ранние правила.

выделить поле ввода, которое содержит активный мигающий курсор:

```
input:focus {
 border: 2px solid red;
 background-color: lightgray;
}
```

Псевдоэлементы

1. позволяют оформить определенные части элементов, а не весь элемент (например, первую букву в этом элементе),
2. позволяют вставлять содержимое перед или после определенных элементов (генерация контента с помощью CSS)

- не поддерживаются в IE 6 (и более младших версиях).
- не могут применяться к внутренним стилям, только к таблице связанных или глобальных стилей.

```
селектор:псевдоэлемент { описание_правил_стиля }
```

- **`first-letter`** – определяет стиль первого символа в тексте элемента, к которому добавляется. Позволяет создавать в тексте буквицу и выступающий инициал.
- **`first-line`** – определяет стиль первой строки блочного текста.
- **`after`** – вставка назначенного контента после элемента, работает совместно со стилевым свойством `content`, который определяет содержимое для вставки. Не поддерживаются браузером Internet Explorer ни в одной его версии.
- **`before`** – по своему действию `before` аналогичен псевдоэлементу `after`, но вставляет контент до элемента.

Псевдоэлементы

- создать буквицу в начале каждого параграфа документа:

```
p:first-letter {
  font-weight: bold;
  font-size: 250%;
  background-color: red;}
```

- сделать первую строку каждого параграфа жирной:

```
p:first-line {
  font-weight: bold;}
```

- вставка декоративных изображений после каждой ссылки на страницу:

```
a:after{ content: " " url(flower.gif);}
```

Можно использовать функцию `attr()` для вставки значений атрибутов элементов после элемента.

- Вставить в скобках целевой адрес каждой ссылки в документе :

```
a:after{  content: "(" attr(href) ")";}
```

Такие правила подходят для таблиц стилей печати.

Сокращенная запись CSS

- Можно объединить несколько связанных свойств CSS в одно свойство.
- правило для полей (сокращения для отступов и границ работают таким же образом):

```
div.foo {
  margin-top: 1px;
  margin-right: 1.5px;
  margin-bottom: 2px;
  margin-left: 2.5px;
}
```

Это правило можно записать короче:

```
div.foo {
  margin: 1px 1.5px 2px 2.5px;
}
```

Задание менее четырех значений для сокращенного свойства

- Одно значение применяется ко всем четырем сторонам

```
margin: 2px;
```

- Первое значение применяется к верху и низу, второе к левому и правому краю

```
margin: 2px 5px;
```

- Первое и третье значения применяются к верху и низу соответственно, второе значение применяется к левому и правому краю

```
margin: 2px 5px 1px;
```

- Значения применяются к верху, правому краю, низу, и левому краю

```
margin: 1px 1.5px 2px 2.5px;
```

Сокращения для шрифта

- определить размер шрифта, толщину, стиль, семейство и высоту строки.

```
font-size: 1.5em;
line-height: 200%;
font-weight: bold;
font-style: italic;
font-family: Georgia, "Times New Roman",
  serif
```

- с помощью одной строки:

```
font: 1.5em/200% bold italic Georgia,"Times
  New Roman",serif;
```

Сокращение для фона

- определить цвет фона, фоновое изображение, повторение изображения и позицию изображения.

```
background-color: #000000;
background-image: url(image.gif);
background-repeat: no-repeat;
background-position: top left;
```

- с помощью одной строки:

```
background:#000000 url(image.gif) no-repeat
  top left;
```

Сокращения для списков

- задать значения типа маркера списка, позиции и изображения.

```
list-style-type: circle;
list-style-position: inside;
list-style-image: url(bullet.gif);
```

- Это эквивалентно следующему:

```
list-style: circle inside url(bullet.gif);
```

Фундаментальные концепции CSS

- **Наследование** связано с тем, как элемент в разметке HTML наследует свойства своих элементов-предков (в которых он содержится) и передает их своим потомкам,
- **Каскадирование** имеет дело с объявлениями CSS, которые применяются к документу, и как конфликтующие правила переопределяют друг друга.

Наследование в CSS

- **Каждый элемент в документе HTML будет наследовать все наследуемые свойства своего предка**, за исключением корневого элемента (html), который не имеет предка.
- Наследуются не все свойства CSS (таблица свойств в Спецификации CSS (http://www.w3.org/TR/CSS21/propidx.html).
- **!** значения, заданные в виде процентных величин, не наследуются.

```
<h1>Этот заголовок <em>очень важен</em>!</h1>
```

Если элементу `em` не присвоен цвет, то он унаследует цвет своего предка, т.е. элемента `h1`. Для задания стиля отображения элементов по умолчанию, достаточно задать стиль элемента `body`.

- Для свойств, которые не наследуются по умолчанию, можно определить принудительное наследование, используя ключевое слово **`inherit`**. (IE не поддерживает)

  Например, следующее правило заставит все параграфы наследовать все свойства фона от своих предков:

```
p {background: inherit;}
```

Каскадирование

- механизм, который управляет конечным результатом, в случае когда несколько конфликтующих объявлений CSS применяются к одному элементу.
- Три основные концепции, которые управляют порядком, в котором применяются объявления CSS:
  1. Важность (наиболее значимая)
  2. Специфичность
  3. Порядок исходного кода

- Если два объявления имеют одинаковую важность, специфичность правил определяет, какое из них будет применяться.
- Если правила имеют одинаковую специфичность, то порядок исходного кода управляет результатом.

Важность

Важность объявления CSS зависит от того, где оно определено. Конфликтующие объявления будут применяться в следующем порядке, более поздние будут переопределять предыдущие:

1. Встроенные таблицы стилей браузера пользователя
2. Обычные объявления в таблицах стиля пользователя (Не все браузеры поддерживают, могут быть очень полезны для пользователей с некоторыми типами функциональных недостатков)
3. Обычные объявления в таблицах стиля автора (разработчика)
4. Важные объявления в таблицах стиля автора
5. Важные объявления в таблицах стиля пользователя

- Для того чтобы превратить обычное объявление в ***важное*** за ним необходимо разместить директиву **!important**.
- Важные объявления в таблице стилей пользователя будут перекрывать все остальное.

```
* { font-family: "Comic Sans MS" !important; }
```

Специфичность

- **Специфичность** определяют как меру того, насколько конкретным является селектор некоторого правила.

- Селектор с низкой специфичностью может соответствовать многим элементам (такой как *, который соответствует каждому элементу в документе), в то время как селектор с высокой специфичностью может соответствовать только одному элементу на странице (такой как `#nav`, который соответствует только элементу с `id` равным `nav`).

- Если два или больше объявлений конфликтуют за данный элемент, и все объявления имеют одинаковую важность, то приоритет в правиле будет иметь объявление с наиболее специфичным селектором.

Порядок исходного кода

- Если два объявления влияют на один и тот же элемент, имеют одинаковую важность и одинаковую специфичность, то окончательное решение определяет порядок исходного кода. **Объявление, которое появляется позже в таблицах стилей будет переопределять те, которые встречаются раньше**.
- Если имеется единственная внешняя таблица стилей, то **объявления в конце файла будут переопределять объявления, которые встречаются раньше в файле**, если возникает конфликт.
- Конфликтующие объявления могут также возникать в различных таблицах стилей. В этом случае **порядок, в котором присоединяются таблицы стилей, включаются или импортируются, управляют тем, какое объявление будет применяться**.

Пример таблицы стилей:

```
h2 {
   font-size: 1.75em;
   color: #469;
}

#container {
   padding: 0;
}

#col_r_content,#col_l_content {
   margin: 10px;
}

p#paragraph1 {
  margin: 0;
}
```

```
#masthead img {
   float: left;
   margin: 0;
   padding: 0;
}

#navigation a:hover {
   color: #000;
   text-decoration: none;
   border: 1px solid #ed9;
   background-color: #ed9;
}

.style_italic {
   font-style: italic;
}
```

CSS-свойства для работы со шрифтами

Свойство	Значение	Описание
font-family	*имя_шрифта*	Задает список шрифтов
font-style	normal italic oblique	Нормальный шрифт Курсив Наклонный шрифт
font-variant	normal small-caps	 Капитель (особые прописные буквы)
font-weight	normal lighter bold bolder 100-900	Нормальная жирность Светлое начертание Полужирный Жирный 100-светлый шрифт, 900-самый жирный
font-size	normal pt px %	Нормальный размер Пункты Пикселы Проценты

CSS-свойства для работы с текстом

Свойство	Значение	Описание
line-height	normal *множитель* *точно* %	Межстрочный интервал
text-decoration	none underline overline line-through blink	Убрать все оформление Подчеркивание Линия над текстом Перечеркивание Мигание текста
text-transform	none capitalize uppercase lowercase	Убрать все эффекты Начинать с прописных Все прописные Все строчные
text-align	left right center justify	Выравнивание текста
text-indent	*Точно* %	Отступ первой строки

CSS-свойства для работы с текстом

color	*Color*	Задает цвет текста
direction	ltr rtl	Направление текста слева направо Направление текста справа налево
letter-spacing	normal *length*	Обычный пробел между словами Фиксированный пробел между словами
text-shadow	none *color* *length*	
white-space	 normal pre nowrap	Задает способ обращения с пробелами внутри элемента Браузер игнорирует пробел Браузер сохраняет пробел. Текст не будет переноситься на другую строку, пока не встретится тег
word-spacing	normal *length*	Увеличивает или уменьшает пробел между словами

CSS-свойства для работы с цветами

Свойство	Значение	Описание
color	*Цвет*	Устанавливает цвет текста
background-color	*Цвет* transparent	Цвет фона Прозрачный
background-image	*URL* none	Фоновый рисунок Отсутствует
background-repeat	repeat repeat-x repeat-y no-repeat	Повторяемость фонового рисунка
background-attachment	scroll fixed	Прокручиваемость фона вместе с документом
background-position	% *Пикселы* top center bottom left right	Начальное положение фонового рисунка

Цвет

- **По его названию.** Браузеры поддерживают некоторые цвета по их названию.
- **По шестнадцатеричному значению.** Цвет можно устанавливать по его шестнадцатеричному значению, как и в обычном HTML.
  - Допустимо использовать сокращенную запись, вроде #fc0. Она означает, что каждый символ дублируется, в итоге получим #ffcc00.
- **С помощью RGB.** Можно определить цвет, используя значения красной, зеленой и синей составляющей в десятичном исчислении. Значение каждого из трех цветов может принимать значения от 0 до 255. Также можно задавать цвет в процентном отношении

Black	#000000	rgb(0,0,0)
Maroon	#800000	rgb(128,0,0)
Green	#008000	rgb(0,128,0)
Olive	#808000	rgb(128,128,0)
Navy	#000080	rgb(0,0,128)
Purple	#800080	rgb(128,0,128)
Teal	#008080	rgb(0,128,128)
Silver	#C0C0C0	rgb (192,192,192)
Gray	#808080	rgb (128,128,128)
Red	#FF0000	rgb(255,0,0)
Lime	#00FF00	rgb(0,255,0)
Yellow	#FFFF00	rgb(255,255,0)
Blue	#0000FF	rgb(0,0,255)
Fuchsia	#FF00FF	rgb(255,0,255)
Aqua	#00FFFF	rgb(0,255,255)
White	#FFFFFF	rgb (255,255,255)

Псевдоклассы для работы с сылками

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.

CSS-свойства для работы со списками

Свойство	Значение	Описание
list-style	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.
list-style-image	none URL	Устанавливает символом маркера любую картинку.
list-style-position	outside inside	Выбор положения маркера относительно блока строк текста.

Единицы измерения

- ***Относительные единицы*** определяют размер элемента относительно значения другого размера. Обычно используют для работы с текстом, либо когда надо вычислить процентное соотношение между элементами.

  em - высота шрифта текущего элемента
  ex - высота символа x
  px – пиксель (элементарная точка, отображаемая монитором или другим подобным устройством)
  % - процент

- ***Абсолютные единицы*** не зависят от устройства вывода, применяются реже, чем относительные и, как правило, при работе с текстом.

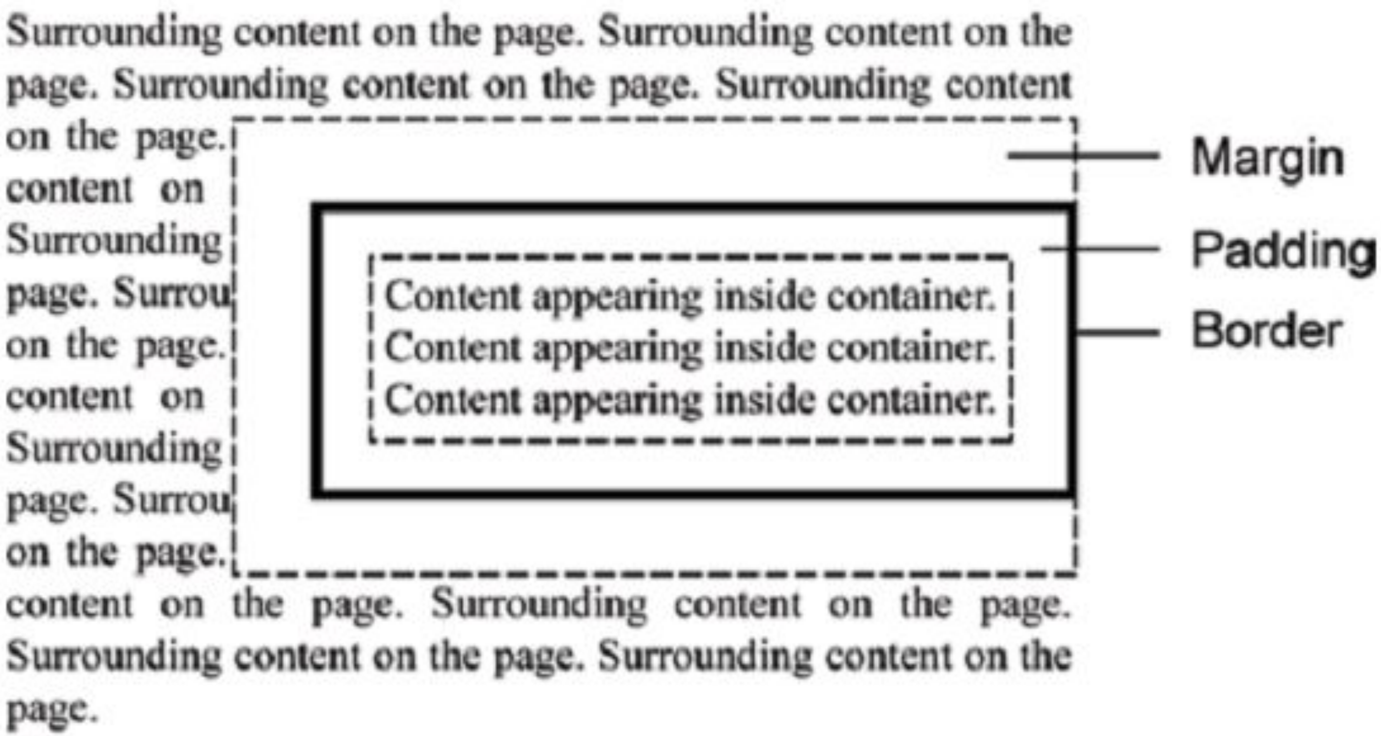
  in - дюйм (1 дюйм равен 2,54 см)
  cm – сантиметр
  mm – миллиметр
  pt - пункт (1 пункт равен 1/72 дюйма)
  pc - пика (1 пика равна 12 пунктам)

Параметры фона

background	Служит для задания всех параметров фона в одном объявлении	background-color background-image background-repeat background-attachment background-position
background-attachment	фиксированное расположение изображения или перемещающееся вместе с остальной страницей	scroll fixed
background-color	Задает цвет фона элемента	color-rgb, color-hex, color-name, transparent
background-image	Задает в качестве фона изображение	url none
background-position	Задает начальное положение фонового изображения	top left, top center, top right, center left, center center, center right, bottom left, bottom center, bottom right, x-% y-%, x-pos y-pos
background-repeat	Определяет, будет ли и каким образом повторяться фоновое изображение	repeat repeat-x repeat-y no-repeat

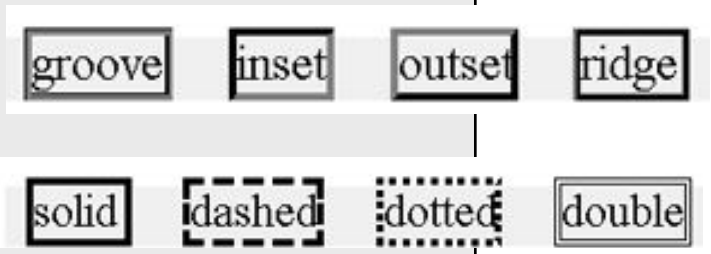
Поля, границы и отступы

- поля (padding)
- границы (border)
- отступы (margin)



Стили и свойства границ

border-style **border-top-style** **border-right-style** **border-bottom-style** **border-left-style**	dashed dotted double groove inset none outset ridge solid
border-width **border-top-width** **border-right-width** **border-bottom-width** **border-left-width**	thin medium thick px
border-color **border-top-color** **border-right-color** **border-bottom-color** **border-left-color**	#000000 - #ffffff color name rgb(r,g,b)
border	border:style size color



Стили и свойства полей

- Для каждой из частей этих компонентов можно задавать одинаковый размер для всех четырех сторон контейнера, или отдельные стороны могут иметь различные значения.

padding	Параметр для задания всех полей в одном объявлении	padding-top padding-right padding-bottom padding-left
padding-bottom	Задает нижнее поле элемента	length %
padding-left	Задает левое поле элемента	length %
padding-right	Задает правое поле элемента	length %
padding-top	Задает верхнее поле элемента	length %

Стили и свойства отступов

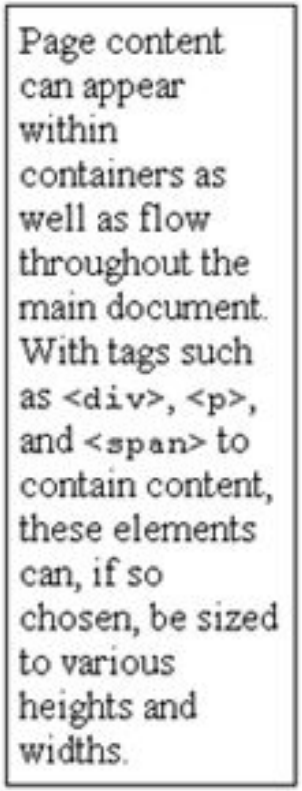
margin	Параметр для задания отступов в одном объявлении	margin-top margin-right margin-bottom margin-left
margin-bottom	Задает нижний отступ элемента	auto length %
margin-left	Задает левый отступ элемента	auto length %
margin-right	Задает правый отступ элемента	auto length %
margin-top	Задает верхний отступ элемента	auto length %

Управление видимостью и положением элементов

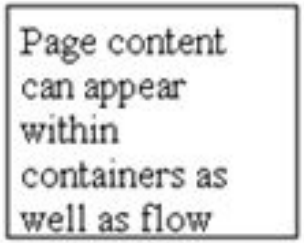
clear	Задает стороны элемента, на которых не допускаются другие перемещаемые элементы	left right both none
cursor	Задает тип выводимого курсора	auto move text wait help ...
display	Определяет, как в документе будет показан элемент	none inline block list-item ...
float	Определяет, где в другом элементе появится изображение или текст (обтекание элементов)	left right none
position	Задает статическое, относительное, абсолютное или фиксированное положение элемента	static relative absolute fixed
visibility	Определяет, будет ли элемент видим или невидим	visible hidden collapse

Задание размеров элементов

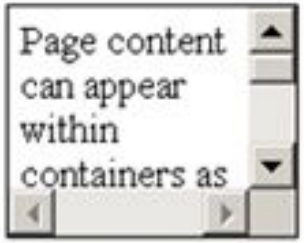
height	px % auto
width	px % auto
overflow	visible hidden scroll auto



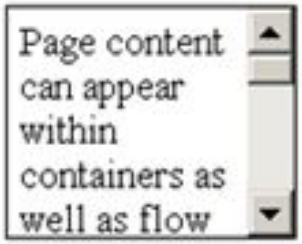

```
width:125px;
height:100px;
overflow:visible
```




```
width:125px;
height:100px;
overflow:hidden
```




```
width:125px;
height:100px;
overflow:scroll
```



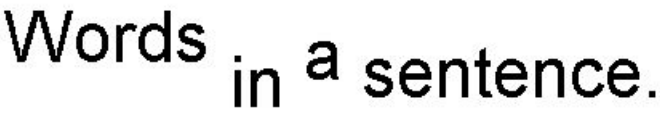

```
width:125px;
height:100px;
overflow:auto
```

Параметры позиционирования в CSS

bottom	Задает, насколько далеко нижний край элемента находится выше/ниже нижнего края родительского элемента	auto % length
clip	Задает форму элемента. Элемент вырезается по форме и выводится.	shape auto
left	Задает, насколько далеко левый край элемента находится правее/левее левого края родительского элемента	auto % length
right	Задает, насколько далеко правый край элемента находится левее/правее правого края родительского элемента	auto % length
top	Задает, насколько далеко верхний край элемента находится выше/ниже верхнего края родительского элемента	auto % length
vertical-align	Задает выравнивание элемента по вертикали	baseline sub super top text-top middle bottom text-bottom length %
z-index	Задает порядковый номер элемента в стеке	auto number

Относительное позиционирование

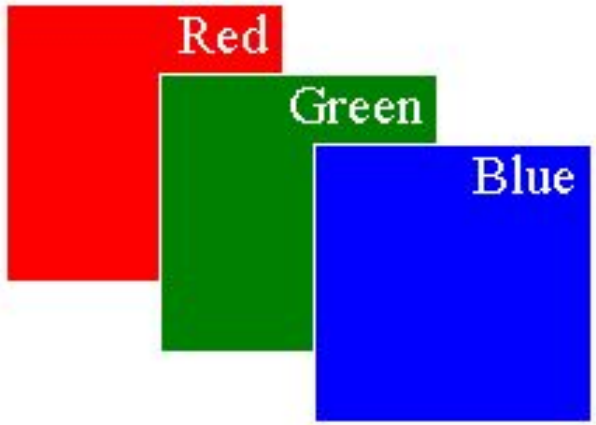
```
<div style="font-size:24pt">

<span style="position:relative;
 top:-15px">Words
</span>

<span style="position:relative;
   top:+10px">in
</span>

<span style="position:relative;
   top:-5px">a
</span>

<span style="position:relative;
   top:+5px">sentence.
</span>
</div>
```



EXAMPLES\css_05.html

Разбиение элементов по слоям

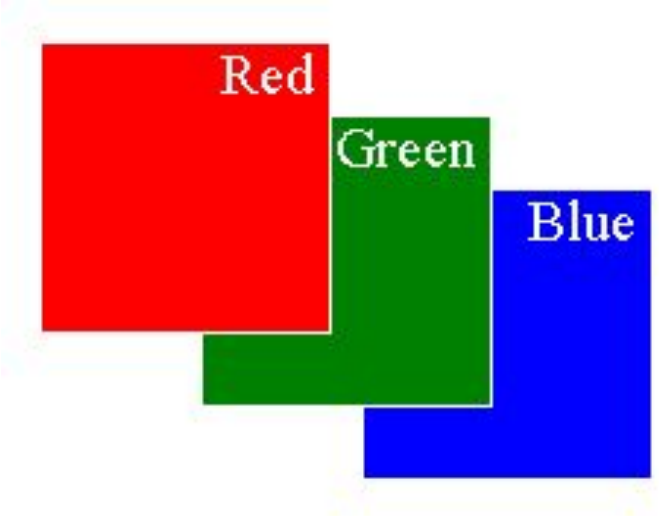
```
<style type="text/css">
  .RED {  position:absolute;
     width:100px; height:100px;
     left:0px; top:0px;
     background-color:red;
     border:solid 1px white;
     color:white;
     text-align:right;}
  .GREEN{position:absolute;
    width:100px; height:100px;
    left:50px; top:25px;
    background-color:green;
    border:solid 1px white;
    color:white;
    text-align:right;}
  .BLUE {position:absolute;
    width:100px; height:100px;
    left:100px; top:50px;
    background-color:blue;
    border:solid 1px white;
    color:white;
    text-align:right;
    }
</style>
...
<body>
   <div class="RED">Red</div>
   <div class="GREEN">Green</div>
   <div class="BLUE">Blue</div>
...
```



EXAMPLES\css_06.html

Разбиение элементов по слоям

```
<style type="text/css">
  .RED {  position:absolute;
     width:100px; height:100px;
     left:0px; top:0px;
     background-color:red;
     border:solid 1px white;
     color:white;
     text-align:right;
     z-index:30;}
  .GREEN{position:absolute;
    width:100px; height:100px;
    left:50px; top:25px;
    background-color:green;
    border:solid 1px white;
    color:white;
    text-align:right;
    z-index:10;}
  .BLUE {position:absolute;
    width:100px; height:100px;
    left:100px; top:50px;
    background-color:blue;
    border:solid 1px white;
    color:white;
    text-align:right;
    z-index:3;
    }
</style>
...
```



EXAMPLES\css_07.html