Основные CSS свойства

Блок №1. Свойство color - цвет текста

Свойство **color** позволяет задать **цвет текста**. Цвет можно задавать следующими способами: английским **словом**, через решетку #, через **rgb** и некоторыми другими способами, которые мы пока не будем разбирать.

Способ первый - английское слово

Самый простой и понятный вариант задать цвет - указать его английским словом, например, **red** задает *красный* цвет, **blue** - *голубой*, **green** - *зеленый*, **black** - *черный*, **white** - *белый*.

В следующем примере всем абзацам на странице задан красный цвет:

```
<p>
    Абзац с текстом.
</p>
```

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

Так код будет выглядеть в браузере:

Абзац с текстом.

Первый способ задания цвета имеет некоторые недостатки. **Во-первых**, таким образом можно сделать не все оттенки цветов (все-таки количество названий ограничено, а оттенков можно придумать огромное множество). **Во-вторых**, невозможно помнить все названия цветов наизусть и, если нужен какой-то редкий цвет, приходится лазить по справочнику и искать его.

Поэтому в CSS существуют и более универсальные способы задать нужный вам цвет. Чтобы понять эти способы, для начала вам необходимо разобраться с тем, как получается нужный цвет на экране компьютера.

На самом деле отдельная точка экрана (*пиксель*) не может светиться всеми цветами, которые нужны, так как это технически было бы очень сложно. И на самом деле каждая точка экрана может светиться только тремя цветами: красным, зеленым и голубым. Причем одновременно и в разных пропорциях.

Комбинируя эти цвета мы можем получить любой нужный нам цвет подобно тому, как это делают художники с красками - если смешать два цвета - мы получим третий.

Способ второй - через rgb

Суть этого способа заключается в следующем - для свойства **color** я пишу следующее значение - *rgb(красный, зеленый, голубой)* - и указываю в каких пропорциях нужно брать эти три базовых цвета. Сами цвета могут изменяться от **0** до **255**. Причем ноль - это отсутствие цвета, а 255 - это чистый цвет (к примеру, чисто красный).

Сами буквы **rgb** расшифровываются как *red* (красный), *green* (зеленый), *blue* (голубой).

А вот так мы получим чисто **красный** цвет (первое значение в 255, а все остальные нули):

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    color: rgb(255, 0, 0);
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Так получим чисто **зеленый** цвет:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    color: rgb(0, 255, 0);
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Способ третий - через

Второй способ получения цвета (через rgb), не смотря на то, что позволяет получить любой оттенок любого цвета, все же является несколько громоздким. Поэтому существует и третий вариант задать цвет - через **шестнадцатеричное** значение.

Чтобы понять суть этого способа, вам нужно разобраться с **шестнадцатеричной** системой счисления. В ней, в отличии от десятичной, который мы с вами пользуемся не 10 цифр, а 16: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F** - как вы видите, за недостатком цифр после девяти **используются буквы**. Преимущество шестнадцатеричной системы в том, что число 255 (и менее) можно представить всего двумя символами - **FF** соответствует **255** и так далее.

В этой системе вместо **rgb()** записывается решетка **#**, после которой идут **6** знаков. Первые 2 знака - красный цвет, вторые два знака - зеленый, и последние два знака - голубой. Цвета также изменяются от **0** до **255**, но в шестнадцатеричной системе это будет от **00** до **FF**.

К примеру, запись **rgb(255, 200, 255)** можно представить как **#FFC8FF** - это будет более компактно.

В следующем примере абзац красится в красный цвет:

```html
<p>
    Абзац с текстом.
</p>
```

```css
p {
    color: #FF0000;
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Мы уже разбирали, что **rgb(0, 0, 0)** соответствует черному цвету, а значит **#000000** - тоже черный цвет, а **rgb(255, 255, 255)** - белый и **#FFFFFF** - тоже белый.

Кроме того, существует сокращенная форма записи - к примеру, **#FEDFED** можно переписать как **#FED** - то есть, если первая половина записи совпадает со второй половиной - то ее (вторую половину) можно не писать. В сокращенном варианте черный цвет будет **#000**, а белый - **#FFF**.

Блок №2. Свойства width и height - ширина и высота

Свойства **width и height** позволяют задать высоту и ширину элементу соответственно. Ширина и высота обычно измеряются в **пикселях** (обозначается **px**) или **процентах** (обозначается **%**) (можно измерять и других единицах, которые мы пока не будем разбирать).

Пиксель - это минимальная точка на экране. При некотором опыте вы сможете легко определять на глаз, сколько пикселей размер у того или иного элемента. Для этого также можно использовать и специальные измерительные инструменты.

Если задавать размеры в **процентах** - то эти проценты вычисляются **относительно родительского элемента**.

В данном примере абзацу задана ширина и высота, а также **граница**, чтобы можно было увидеть, где заканчивается наш абзац:

```html
<p>
    Абзац с текстом.
</p>
```

```css
p {
    width: 300px;
    height: 100px;
    border: 1px solid black;
}
```

Так код будет выглядеть в браузере:

Блок №3. Свойство text-align - выравнивание текста

Свойство **text-align** позволяет задать выравнивание текста. Текст можно выровнять по **левому** краю (значение **left**), по **правому** (значение **right**), по **центру** (значение **center**) и **одновременно** и по правому, и по левому краю (значение **justify**).

Давайте посмотрим на примерах, что имеется ввиду.

Значение left

Давайте сделаем так, чтобы текст был выровнен по **левому** краю. Для этого нужно свойство **text-align** поставить в значение **left**:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    text-align: left;
    width: 300px;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

Для абзацев значение **left** можно и не ставить - они по умолчанию и так выровнены по **левому** краю. Однако есть элементы, которые по умолчанию стоят **по центру** (это, например, теги **th**, которые делают ячейку-заголовок таблицы). И вот для них как раз-таки и может потребоваться выравнивание по левой стороне.

Значение right

Давайте теперь поставим текст по **правому** краю

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    text-align: right;
    width: 300px;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

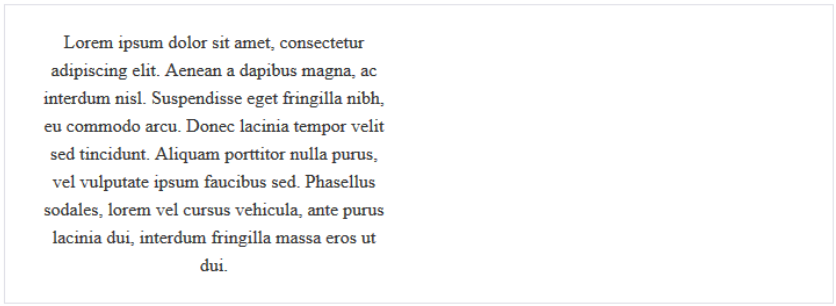
Значение center

Поставим текст **по центру**:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    text-align: center;
    width: 300px;
}
```

Так код будет выглядеть в браузере:



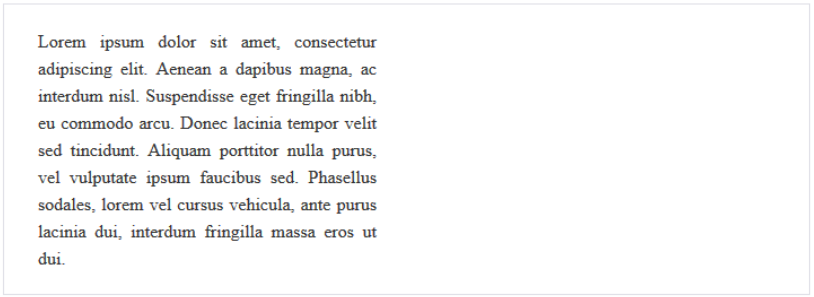
Значение justify

Ну, и наконец выровняем текст одновременно и по правому, и по левому краю:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    text-align: justify;
    width: 300px;
}
```

Так код будет выглядеть в браузере:



Блок №4. Свойство font-weight - жирность

Свойство **font-weight** позволяет сделать текст жирным или наоборот - отменить жирность (к примеру, для заголовков, которые жирные по умолчанию). Чтобы сделать текст жирным, следует добавить значение **bold**, а чтобы отменить жирность - значение **normal**.

Это свойство работает аналогично тегу **b**, который мы с вами уже разобрали. Разница в том, что через CSS управлять жирностью гораздо удобнее - можно заставить все абзацы стать жирными, а все заголовки - нежирными, сделав всего пару записей в CSS файле.

В следующем примере мы сделаем все абзацы жирными, а все заголовки h3 - нежирными:

```html
<h3>Это заголовок</h3>
<p>
    Абзац с текстом.
</p>
```

```css
h3 {
    font-weight: normal;
}

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

Так код будет выглядеть в браузере:

Это заголовок

Абзац с текстом.

Для сравнения привожу их вид по умолчанию:

Это заголовок

Абзац с текстом.

Блок №5. Свойство font-style - курсив

Свойство **font-style** позволяет сделать текст курсивным или наоборот - отменить курсив. Чтобы сделать текст курсивом, следует добавить значение **italic**, а чтобы отменить курсив - значение **normal**. Смотрите пример:

```html
<p>
    Абзац с текстом.
</p>
```

```css
p {
    font-style: italic;
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Блок №6. Свойство font-size - размер текста

Свойство **font-size** позволяет задать размер текста. Размер задается в **пикселях** (обозначаются **px**), в **пунктах** (обозначаются **pt**), в **процентах** (обозначаются **%**) и с помощью некоторых других единиц, которые мы пока не будем разбирать.

Значение в пунктах

Пункты обозначаются в программе Word. Там вы можете задать, к примеру, шрифт размером 16 - это число и есть размер в пунктах. При этом этот шрифт в Ворде будет такого же такого же размера, как и шрифт 16 пунктов в браузере.

Между пунктами и пикселями есть соответствие: **12pt = 16px**. При этом все размеры на экране на самом деле меряются в пикселях, даже если вы задаете их в пунктах. При этом, если после перевода пунктов в пиксели, полученные пиксели будут дробными - они округлятся до ближайшего целого.

Давайте зададим тексту абзаца шрифт в 20 пунктов:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    font-size: 20pt;
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Блок №7. Свойство font-family - тип шрифта

Свойство **font-family** позволяет задать тип шрифта (тип часто называют *семейством* шрифта).

Поставим для всех абзацев шрифт **Arial**:

```
<p>
    Абзац с текстом.
</p>
```

```
p {
    font-family: Arial;
}
```

Так код будет выглядеть в браузере:

Абзац с текстом.

Не все шрифты можно использовать, так как если на компьютере пользователя не окажется указанного шрифта - браузер вместо него возьмет стандартный шрифт (в результате на экране будет совсем не то, что мы задумывали). Поэтому необходимо использовать только **веб безопасные шрифты**, которые наверняка будут у пользователя (их список будет чуть ниже).

Также для решения данной проблемы поступают так: перечисляют несколько похожих шрифтов через запятую. Например, **font-family: Georgia, "Times New Roman"**.

Когда браузер встречает первый шрифт в списке, он проверяет его наличие на компьютере пользователя. Если такого шрифта нет, берется следующий шрифт из списка и также анализируется на присутствие. Поэтому несколько шрифтов увеличивает вероятность, что хотя бы один из них будет обнаружен на клиентском компьютере.

Заканчивают список обычно ключевым словом, которое описывает тип шрифта (все шрифты относятся к какому-нибудь типу) — **serif, sans-serif, cursive, fantasy** или **monospace**. Если

браузер не нашел ни одного из указанных шрифтов на компьютере пользователя, то он выберет один из шрифтов указанного типа.

Шрифты бывают с засечками **serif** и без засечек **sans-serif**. **Засечки** - это специальные штрихи на концах букв (выделены красным):

Засечки являются отличительной особенностью шрифта, нет такого, чтобы был одновременно с засечками и без засечек. **Arial с засечками** будет уже какой-то другой шрифт, но никак не Arial.

Если в имени шрифта содержатся пробелы, например, Times New Roman, оно должно заключаться в одинарные или двойные кавычки.

Веб безопасные шрифты

Безопасным шрифтом можно назвать такой шрифт, который является стандартным для всех операционных систем. Поскольку о таком положении дел остается только мечтать, то абсолютно безопасных шрифтов не существует! Отдельные шрифты можно назвать безопасными с некоторыми оговорками. Вот они:

Значение	Описание
Arial	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.
Arial Black	**Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.**
Comic Sans MS	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.
Courier New	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.
Georgia	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.
Impact	**Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.**
Times New Roman	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.
Trebuchet MS	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a

	dapibus magna, ac interdum nisl.
Verdana	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl.

Блок №8. Свойство line-height - межстрочный интервал

Свойство **line-height** задает межстрочный интервал.

Межстрочный интервал - это расстояние между линиями текста, то есть белый промежуток между ними.

При использовании свойства **line-height** вас может ожидать некоторый **подвох**: это свойство не задает промежуток между строками текста, как могло бы показаться, а задает **высоту линии текста**.

Это значит, что реальный видимый промежуток между строками будет вычисляться так: **line-height - font-size** = **расстояние между строками текста**. Или наоборот **line-height** = **font-size** + **расстояние между строками текста**.

В данном примере расстояние между строками текста будет **line-height - font-size = 50px - 13px = 37px**:

```
<p>
    Длинный текст...
</p>
```

```
p {
    font-size: 13px;
    line-height: 50px;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

Значением свойства **line-height** не обязательно должно служить число в некоторых единицах. Можно также просто написать число или дробь. В этом случае настоящее значение **line-height** можно будет найти, умножив его на значение **font-size**.

К примеру, **font-size** равен **10px**, а **line-height - 1.5**. В этом случае реальное значение **line-height** будет **10px * 1.5 = 15px**. Ну, а видимый белый промежуток между линиями текста будет 5px: **line-height - font-size = 15px - 10px = 5px**.

Преимущество такого способа задания **line-height** в том, что при изменении размера шрифта автоматически будет меняться и межстрочный интервал.

Блок №8. Свойство font - сокращение для шрифтов

Существует специальное свойство **font**, которое можно использовать вместо многих свойств, которые мы уже разобрали. Такие свойства в CSS называются **свойствами-сокращениями**. Зачастую их использование гораздо удобнее вместо множества других свойств.

Свойство **font** имеет следующий синтаксис: **курсив жирность размер_шрифта/интервал_между_строками семейство_шрифта**. Обязательными являются "размер_шрифта" и "семейство_шрифта", порядок имеет значение.

Свойство **font** является **свойством-сокращением**.

```
<p>
    Длинный текст...
</p>
```

```
p {
    font: 16px Arial;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

```
<p>
    Длинный текст...
</p>
```

```
p {
    font: italic 16px/50px Arial, sans-serif;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

Блок №9. Свойство text-indent - красная строка

Свойство **text-indent** позволяет задать **красную строку**, то есть отступ первой строки текста (к примеру, в абзаце).

Давайте сделаем красную строку для абзацев:

```html
<p>
    Длинный текст...
</p>

<p>
    Длинный текст...
</p>

<p>
    Длинный текст...
</p>
```

```css
p {
    text-indent: 50px;
}
```

Так код будет выглядеть в браузере:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean a dapibus magna, ac interdum nisl. Suspendisse eget fringilla nibh, eu commodo arcu. Donec lacinia tempor velit sed tincidunt. Aliquam porttitor nulla purus, vel vulputate ipsum faucibus sed. Phasellus sodales, lorem vel cursus vehicula, ante purus lacinia dui, interdum fringilla massa eros ut dui.