



CSS

Содержание

| | |
|--|--------------|
| 1. Подключение стилей | 3-4 |
| 2. Основные единицы измерения CSS | 5-8 |
| 3. Цвет в CSS | 9-11 |
| 4. Селекторы | 12-20 |
| • Селекторы элементов | 13 |
| • Селекторы класса | 13 |
| • Селекторы идентификаторов | 14 |
| • Селекторы потомков | 14 |
| • Селекторы дочерних элементов | 15 |
| • Универсальные селекторы | 16 |
| • Селекторы соседних элементов | 16 |
| • Селекторы атрибутов | 16 |
| • Селекторы псевдоэлементов | 17 |
| • Селекторы псевдоклассов | 18 |
| • Псевдокласс :not | 19 |
| • Псевдокласс :nth-child | 19 |
| • Псевдокласс :nth-of-type | 20 |
| 5. Правила специфичности | 21-23 |
| 6. Основные свойства CSS | 24-36 |
| • Работа с цветом и фоном | 25 |
| • Работа со шрифтом | 27 |
| • Работа с текстом | 29 |
| • Разрыв строк | 32 |
| • Вертикальное выравнивание фрагмента | 33 |
| • Свойства рамки | 34 |
| • Свойства списков | 35 |

1

Подключение стилей

CSS-стили можно писать внутри HTML-кода страницы или подключать их как внешний файл.

В первом случае стили называются *«встроенными»* или *«инлайновыми»*, а писать их нужно внутри тега **<style>**. Этот тег обычно размещают внутри **<head>**. Например:

```
<head>
  <style>
    CSS-код
  </style>
</head>
```

Внутри **<style>** пишут обычный CSS-код.

Инлайновые стили используют не так часто, например, для оптимизации скорости загрузки страницы. Чаще используют *внешние* стили, которые подключают из внешнего файла с расширением *.css*. Для этого используется тег **<link>**. Например:

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

В атрибуте **href** задают адрес файла, а атрибут **rel = "stylesheet"** говорит браузеру, что мы подключаем стили, а не что-то другое.

Лучше подключать стили внутри **<head>**, но это необязательно. Тег **<link>** будет работать и в другом месте страницы.

2

Основные единицы измерения CSS

Пиксель px – это самая базовая, абсолютная и окончательная единица измерения.

Количество пикселей задаётся в настройках разрешения экрана, один px – это как раз один такой пиксель на экране. Все значения браузер в итоге пересчитает в пиксели.

Наиболее частоиспользуемые единицы в верстке:

| Единица измерения | Обозначение |
|---|-------------|
| Абсолютные единицы | |
| Пиксели | px |
| Относительные единицы измерения | |
| Размер литеры 'm' текущего шрифта | em |
| Размер литеры 'm' корневого элемента | rem |
| Проценты | % |
| 1% от ширины области просмотра | vw |
| 1% от высоты области просмотра | vh |
| 1% от меньшего значения из ширины и высоты области просмотра | vmin |
| Определяется, что больше, значение ширины или высоты области просмотра и от него вычисляется 1% | vmax |

Разница между **em** и **rem** следующая. **em** зависит от размера шрифта родителя элемента и меняется вместе с ним, а **rem** привязан к корневому элементу, т. е. размеру шрифта заданного для элемента html.

Что ещё надо знать? Когда мы говорим «процент», то возникает вопрос – «Процент от чего?». Как правило, это процент от значения свойства родителя с тем же названием, но не всегда.

Вот примеры-исключения, в которых % берётся иначе:

margin-left

При установке свойства **margin-left** в %, процент берётся от ширины родительского блока, а вовсе не от его margin-left.

line-height

При установке свойства **line-height** в %, процент берётся от текущего размера шрифта, а вовсе не от line-height родителя.

width/height

Для **width/height** обычно процент от ширины/высоты родителя, но при position:fixed, процент берётся от ширины/высоты окна (а не родителя и не документа).

Значения **vw, vh, vmin, vmax** были созданы, в первую очередь, для поддержки мобильных устройств. Их основное преимущество – в том, что любые размеры, которые в них заданы, автоматически масштабируются при изменении размеров окна.

Устаревшие или неиспользуемые единицы измерения:

| Единица измерения | Обозначение |
|-----------------------------------|-------------|
| Пункты | pt |
| Размер литеры 'x' текущего шрифта | ex |
| Дюймы | in |
| Сантиметры | cm |
| Миллиметры | mm |

Помните: При установке размеров обязательно указывайте единицы измерения, например **width: 30px**. В противном случае браузер не сможет показать желаемый результат, поскольку не понимает, какой размер вам требуется. Единицы не добавляются только при нулевом значении (**margin: 0**).

3

Цвет в CSS

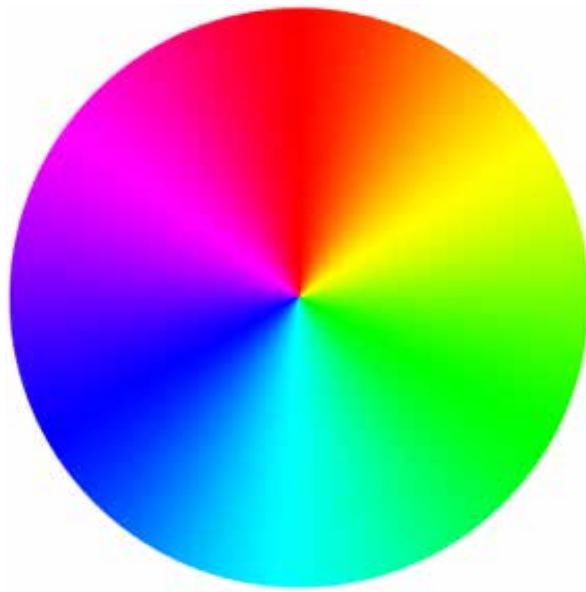
Цвет в стилях можно задавать разными способами: по шестнадцатеричному значению, по названию, в формате RGB, RGBA, HSL, HSLA.

rgb(r.g.b) – Можно определить цвет, используя значения красной, зеленой и синей составляющей в десятичном исчислении. Каждая из трех компонент цвета принимает значение от 0 до 255. Также допустимо задавать цвет в процентном отношении, при этом 100% будет соответствовать числу 255.

#rrggbb – Для задания цветов используются числа в шестнадцатеричном коде. Цифры будут следующие: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Цифры от 10 до 15 заменены латинскими буквами. Чтобы не возникало путаницы в определении системы счисления, перед шестнадцатеричным числом ставят символ решетки #, например #666999. Каждый из трех цветов — красный, зеленый и синий — может принимать значения от 00 до FF. Таким образом, обозначение цвета разбивается на три составляющие #rrggbb, где первые два символа отмечают красную компоненту цвета, два средних — зеленую, а два последних — синюю.

#rgb – Допускается использовать сокращенную форму вида #rgb, где каждый символ следует удваивать. Так, запись #fe0 следует расценивать как #ffee00. Браузеры поддерживают некоторые цвета по их названию.

HSL – Название формата HSL образовано от сочетания первых букв Hue (оттенок), Saturate (насыщенность) и Lightness (светлота). Оттенок – это значение цвета на цветовом круге и задаётся в градусах. 0° соответствует красному цвету, 120° — зелёному, а 240° — синему. Значение оттенка может изменяться от 0 до 359.



Насыщенностью называется интенсивность цвета, измеряется в процентах от 0% до 100%. Значение 0% обозначает отсутствие цвета и оттенок серого, 100% максимальное значение насыщенности.

Светлота задает, насколько цвет яркий и указывается в процентах от 0% до 100%. Малые значения делают цвет темнее, а высокие светлее, крайние значения 0% и 100% соответствуют чёрному и белому цвету.

HSLA. Формат HSLA похож по синтаксису на HSL, но включает в себя альфа-канал, задающий прозрачность элемента. Значение 0 соответствует полной прозрачности, 1 — непрозрачности, а промежуточное значение вроде 0.5 — полупрозрачности.

Пример:

```
color: hsla(220,100%,50%, 0.5);
```

4

Селекторы

Селекторы элементов

Наиболее простой для понимания вид селекторов, при его использовании стиль будет применен ко всем встречающимся в html-документе соответствующим элементам.

Пример:

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

Селекторы класса

В отличие от селекторов элементов, стили, описанные при помощи селектора классов будут применены только к элементам, содержащим атрибут class с соответствующим значением.

Пример:

```
.clrRed {color: red}
```

Селекторы элементов и классов могут быть объединены:

```
p.clrRed {color: red}  
li.clrRed {color: yellow}
```

Таким образом, содержимое тегов ****, относящихся к классу **clrRed**, будет представлено в желтом цвете, содержимого тега **<p>** аналогичного класса – красным.

Также к элементу могут быть применены стили нескольких классов, например:

```
.txtAlign {text-align: justify}  
.txtColor {color: red}  
<p class="txtAlign txtColor"> .. </p>
```

Селекторы идентификаторов

Стиль, описанный при помощи селекторов идентификаторов, может быть применен только к одному элементу веб-страницы. При этом элемент должен содержать атрибут id с соответствующим значением.

Пример оформления селектора идентификаторов:

```
#clrRed {color: red}
```

Как и селекторы классов, селектор идентификаторов может быть объединен с селектором элементов:

```
li#clrRed {color: red}
```

Селекторы потомков

Селектор потомков используется для задания стиля элементов, являющихся потомками другого элемента.

Оформляется селектор потомков следующим образом:

```
элемент-предок      элемент-потомок      { атрибуты стиля }
```

Следующий пример задаст стиль только для элементов ****, являющихся потомком элемента **<div>**:

```
div span {color: red}
```

Селекторы потомков можно объединить с селекторами классов:

```
div.content span {color: red}
```

с селекторами идентификаторов:

```
div#wrap span {color red}
```

или

```
div span#text {color: red}
```

Селекторы дочерних элементов

Стили, описанные при помощи данных селекторов применимы только к дочерним элементам.

Оформляется стиль с селектором дочерних элементов следующим образом:

```
элемент-родитель > дочерний элемент {атрибуты стиля}
```

Как и в предыдущих случаях, допустимо комбинировать различные селекторы:

```
table#red td.txt > span.text {color: red}
```

В этом случае стиль будет применен к содержимому элемента ****,

относящегося к классу **text** и являющемуся дочерним элементом по отношению к тегу **<td>**, который, в свою очередь должен относиться к классу **txt**. При этом указанный тег **<td>** должен быть потомком **<table>** с идентификатором **red**.

Универсальные селекторы

Стили с универсальными селекторами применяются ко всем элементам веб-страницы. Оформление при этом выглядит следующим образом:

```
* {color: red}
```

Селекторы соседних элементов

При помощи данных селекторов задаются стили для элементов, располагающихся сразу же за другим элементом.

Пример:

```
span + p {color: red}
```

Селекторы атрибутов

Селекторы атрибутов, как понятно из их названия, привязываются к элементу, имеющему соответствующий атрибут.

Оформляются данные селекторы следующим образом:

```
элемент [атрибут= "значение атрибута"] {атрибут стиля }
```


Рассмотрим на примере. Следующий стиль **a[href="about.htm"]** будет применен только к содержимому тега **<a>**, значение атрибута **href** которого соответствует **about.htm**.

Селекторы атрибутов не обязательно должны содержать значение атрибута. Стиль для элементов, значения атрибутов которых содержат определенное слово:

```
div[data~="base"] {border: 1px double black}
```

В этом случае стиль будет применен к любому тегу **<div>**, атрибут **data** которого содержит слово **base**.

Селектор по атрибуту, содержащему разделенное дефисом значение:

```
div[title|="base"] {border: 1px double black}
```

Стиль будет применен к любому тегу **<div>**, значение **title** которого содержит **"base-"**.

Селекторы псевдоэлементов

Псевдоэлементы позволяют привязать стиль к определенному фрагменту веб-страницы, например, к первой букве текста элемента веб-страницы. Доступны следующие псевдоэлементы:

| | |
|---------------------|---|
| after | позволяет добавить контент после указанного элемента; |
| before | позволяет добавить контент до указанного элемента; |
| first-letter | задает стиль для первого символа в тексте элемента; |
| first-line | задает стиль первой строки текста элемента. |

Пример оформления:

```
li: before {content: "some content"}
```

Селекторы псевдоклассов

Псевдоклассы позволяют применять различные стили к элементам, в зависимости от их состояния. Например, к ссылке могут быть применены различные стили в зависимости от того, является ли ссылка еще не посещенной, находится ли в "фокусе" курсора.

Доступны следующие "состояния" элементов:

| | |
|---------------------|---|
| :active | стиль применяется к элементу, активированному пользователем; |
| :link | стиль применяется к непосещённым ссылкам; |
| :focus | стиль применяется к элементу при получении им фокуса; |
| :hover | стиль активизируется, когда курсор находится в пределах элемента; |
| :visited | стиль применяется к посещённым гиперссылкам; |
| :last-child | задаёт стилевое оформление последнего элемента своего родителя; |
| :first-child | стиль применяется к первому дочернему элементу селектора. |

Пример:

```
a:link {color: red}  
div:hover {background-color: #cc00ff;}
```

Псевдокласс :not

Псевдокласс **:not(селектор)** является отрицающим селектором. С его помощью можно выбрать элементы, которые **не** содержат указанный селектор. Например, селектор:

```
li:not(:last-child){}
```

выберет все тэги ****, **не** являющиеся последними в их родителе.

В качестве селектора могут указываться псевдоклассы, теги, идентификаторы, классы и селекторы атрибутов. Конструкция **:not(:not(...))** запрещена.

Псевдокласс :nth-child

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

```
элемент:nth-child(odd | even | <число> | <выражение>) {...}
```

Значения:

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.

выражение – Задается в виде $an+b$, где a и b целые числа, a и n — счетчик, который автоматически принимает значение 0, 1, 2...

Если a равно нулю, то оно не пишется и запись сокращается до b . Если b равно нулю, то оно также не указывается и выражение записывается в форме a^n .

Псевдокласс `:nth-of-type`

Псевдокласс **`:nth-of-type`** используется для добавления стиля к элементам указанного типа на основе нумерации в дереве элементов.

Селектор: `:nth-of-type(odd | even | <число> | <выражение>) {...}`

Значения такие же как и у псевдокласса **`:nth-child`**

Результат для различных значений псевдокласса

| Значение | Номера элементов | Описание |
|----------|------------------|---|
| 1 | 1 | Первый элемент. |
| 5 | 5 | Пятый элемент. |
| $2n$ | 2, 4, 6, 8, 10 | Все чётные элементы, аналог значения even . |
| $2n+1$ | 1, 3, 5, 7, 9 | Все нечётные элементы, аналог значения odd . |
| $3n+2$ | 2, 5, 8, 11, 14 | — |
| $5n-2$ | 3, 8, 13, 18, 23 | — |
| even | 2, 4, 6, 8, 10 | Все чётные элементы. |
| odd | 1, 3, 5, 7, 9 | Все нечётные элементы. |

5

Правила специфичности

Специфичность селекторов (selector's specificity) определяет их приоритетность в таблице стилей. Чем специфичнее селектор, тем выше его приоритет.

Существует 4 правила, по которым вычисляется специфичность селекторов:

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`).

Все 4 правила сводятся в одну систему a-b-c-d (где a – наивысший приоритет) и образуют специфичность.

Пример:

1. `p {/*какие-то определения */}`
2. `div p {/*какие-то определения */}`
3. `p.note {/*какие-то определения */}`
4. `form.feedbackForm input[type="text"] {/*какие-то определения */}`
5. `#conten a:hover {/*какие-то определения */}`

| | |
|-------------------------|--|
| Первая строка | одинокий селектор типа. Специфичность 0001. |
| Вторая строка | два селектора типа. Специфичность 0002. |
| Третья строка | селектор типа и класса. Специфичность 0011. |
| Четвёртая строка | два селектора типа, один класса и один атрибута. Специфичность 0022. |
| Пятая строка | селектор идентификатора, типа и псевдокласс. Специфичность 0111. |

Сравниваются специфичности очень просто. Какое число больше, то определение и выиграло. Рекомендуем статью [«Понимание веса CSS-селекторов»](#)

6

Основные свойства CSS

Работа с цветом и фоном

color

задает цвет переднего плана (**color: #00FF00**);

background-color

задает цвет фона элемента (**background-color: brown**);

background-image

задает фоновое изображения для элемента

(**background-image: url("image.gif")**);

background-repeat

задает тип повторения изображения, установленного при помощи атрибута стиля background-image (**background-repeat: no-repeat**), может принимать следующие значения:

| | |
|------------------|--|
| repeat-x | изображение повторяется по горизонтали |
| repeat-y | изображение повторяется по вертикали |
| repeat | изображение повторяется по горизонтали и вертикали |
| no-repeat | изображение не повторяется (значение по умолчанию) |

background-attachment

определяет будет ли фоновое изображение прокручиваться вместе с элементом (background-attachment: fixed), может принимать следующие значения:

| | |
|---------------|---|
| scroll | изображение будет прокручиваться вместе с элементом |
| fixed | прокрутка изображения заблокирована |

background-position

определение координат позиционирования фонового изображения, содержит два значения: положение по горизонтали и положение по вертикали (**background-position: 5cm 4cm**). Помимо числовых, может принимать следующие значения:

| | |
|---------------|---|
| left | горизонтальное позиционирование "по левому краю" |
| center | горизонтальное позиционирование "по центру" |
| right | горизонтальное позиционирование "по правому краю" |
| top | вертикальное позиционирование "сверху" |
| center | вертикальное позиционирование "по центру" |
| bottom | вертикально позиционирование "снизу" |

Работа со шрифтом

font-family

задает семейство используемого шрифта (**font-family: arial**) Для задания шрифта может быть использовано два типа имен: имя семейства (**family-name**) и родовое имя (**generic family**). К именам семейства относятся, собственно, названия шрифтов (**Camria, Arial и т.д.**) Количество родовых имен поскромнее:

| | |
|-------------------|---------------------|
| serif | шрифты с засечками |
| sans-serif | рубленные шрифты |
| cursive | курсивные шрифты |
| fantasy | декоративные шрифты |
| monospace | моноширинные шрифты |

font-style

задает стиль шрифта (**font-style: normal**). Соответственно, принимает значения:

| | |
|----------------|-----------------|
| normal | обычный шрифт |
| italic | курсивный шрифт |
| oblique | наклонный шрифт |

font-variant

задает тип представления строчных букв (**font-variant: normal**). Принимает следующие значения:

| | |
|-------------------|--|
| normal | строчные буквы представляются в исходном регистре |
| small-caps | строчные буквы модифицируются в заглавные, но меньшего размера |

font-weight

определяет насыщенность шрифта (**font-weight: bold**). Принимает следующие значения:

| | |
|---------------|---------------------------------|
| normal | стандартная насыщенность шрифта |
| bold | полужирное начертание |

Ряд браузеров поддерживает числовые значения насыщенности шрифта в пределах от **100** до **900**, где **100** – сверхсветлое насыщение шрифта, **400** – стандартное, **900** – полужирное.

font-size

определяет размер шрифта (**font-size: 12pt**). Может быть представлен в виде констант, абсолютных, или относительных значений.

Работа с текстом

text-align

определяет горизонтальное выравнивание текста элемента (**text-align: center**). Может принимать следующие значения:

| | |
|----------------|---|
| center | выравнивание по центру |
| left | выравнивание по левому краю |
| right | выравнивание по правому краю |
| justify | выравнивание по ширине |
| auto | тип выравнивания не изменяется |
| start | в случае, если направление текста слева направо, то выравнивает по левому краю; если направление текста справа налево – по правому краю |
| end | в случае, если направление текста слева направо, то выравнивает по правому краю; если направление текста справа налево – по левому краю |

text-align-last

задает тип выравнивания последней строки элемента при условии, что значение атрибута стиля **text-align** равно **justify (text-align-last: left)**. Принимает значения аналогичные атрибуту **text-align**.

text-decoration

добавляет эффекты для текста (**text-decoration: none**). Может принимать следующие значения:

| | |
|---------------------|-----------------------------------|
| blink | мигающий текст |
| line-through | зачеркнутый текст |
| overline | линия над текстом |
| underline | линия под текстом (подчеркивание) |
| none | эффектов нет |

text-indent

задает величину отступа для первой строки текста (**text-indent: 10%**). Могут быть указаны конкретные значения и процентные.

text-overflow

задание параметра видимости текста (**text-overflow: clip**). Может принимать два значения:

| | |
|-----------------|---|
| clip | текст обрезается, если выходит за границы элемента |
| ellipsis | при выходе текста за границы добавляется многоточие |

text-shadow

добавляет тень тексту и определяет ее параметры (text-shadow: red 5 5). Могут быть заданы следующие параметры тени:

| | |
|-----------------------------|---|
| none | тени нет |
| цвет | любой поддерживаемый цвет |
| сдвиг по горизонтали | положительное значение сдвигает тень вправо, отрицательное – влево |
| сдвиг по вертикали | положительное значение опускает тень относительно текста, отрицательное поднимает |
| радиус размытия | большее значение сглаживает тень, по умолчанию параметр равен 0. |

text-transform

преобразование текста в заглавные или прописные буквы (text-transform: lowercase). Принимает следующие значения:

| | |
|-------------------|--|
| none | символы не меняются |
| capitalize | первая литера каждого слова становится заглавной |
| lowercase | символы текста преобразовываются в нижний регистр |
| uppercase | символы текста преобразовываются в верхний регистр |

Разрыв строк

white-space

задает способ отображения пробелов между словами (**white-space: pre**).

Принимает следующие значения:

normal

несколько пробелов преобразуются в один, символы перевода строк также преобразуются в пробелы, браузер самостоятельно разрывает текст и переводит его на новые строки

nowrap

несколько пробелов преобразуются в один, символы перевода строк также преобразуются в пробелы, браузер не осуществляет разрыв и перевод строк

pre

последовательность пробелов сохраняется, символы перевода строк также сохраняются, браузер самостоятельно не выполняет разрыв и перенос строк. Фактически, текст выглядит образом, определенным разработчиком, без каких-либо изменений

pre-line

несколько пробелов преобразуются в один, символы перевода строк сохраняются, браузер самостоятельно разрывает текст и переводит его на новые строки

pre-wrap

последовательность пробелов сохраняется, символы перевода строк также сохраняются, браузер самостоятельно выполняет разрыв и перенос строк

word-wrap

указывает места, где браузер может осуществить перевод строки (**word-wrap: normal**). Может принимать следующие значения:

| | |
|-------------------|--|
| normal | строки разрываются только по пробелам |
| break-word | браузер может выполнять разрыв строк внутри слов |

Вертикальное выравнивание фрагмента

При необходимости смещения по вертикали определенного элемента относительно текста применяется атрибут стиля **vertical-align**, принимающий значения:

baseline

выравнивание базовой линии по соответствующей линии родительского элемента. В случае с ячейкой таблицы происходит выравнивание по базовой линии первой текстовой строки;

bottom

выравнивание элемента по нижней части родительского элемента. В случае с ячейкой таблицы происходит выравнивание по нижнему краю;

middle

выравнивание по центру родительского элемента. В случае с ячейкой таблицы происходит выравнивание по середине;

sub

выравнивание базовой линии элемента по базовой линии нижнего индекса родительского элемента;

super

выравнивание базовой линии по базовой линии верхнего индекса родительского элемента;

text-bottom

выравнивание нижнего края фрагмента по нижнему краю текста родителя;

text-top

выравнивание верхнего края фрагмента текста по верхнему краю текста родителя;

top

выравнивание верхнего края фрагмента по верхнему краю текста родителя. В случае с ячейкой таблицы происходит выравнивание по верхнему краю.

Свойства рамки

border

рамка. Имеет толщину, цвет, фактуру и местоположение. Обычно пишется таким образом:

```
border: 1px solid #333;
```

запись означает, что рамка темно-серого цвета, сплошная, толщиной в 1 пиксель. Другие значения фактуры: **dotted** – точечная, **dashed** – пунктирная, **double** – двойная (у этой толщина должна быть никак не меньше 3 пикселей, иначе выйдет одинарная).

Местоположение рамки также легко обозначить в правилах:

| | |
|-------------------|--------|
| border-top | вверху |
|-------------------|--------|

| | |
|----------------------|-------|
| border-bottom | внизу |
|----------------------|-------|

| | |
|--------------------|-------|
| border-left | слева |
|--------------------|-------|

| | |
|---------------------|--------|
| border-right | справа |
|---------------------|--------|

Можно задать различные цвет или толщину рамки сразу для всех 4 сторон объекта. Например, запись

```
border-color: #ccc #f4f5f7 #333 #000;
```

означает, что цвет верхней рамки светло-серый (**#ccc**), справа **#f4f5f7**, снизу **#333**, слева **#000**. Точно так же можно задать и толщину.

Свойства списков

Задается свойство следующим правилом: **list-style-type:**

У маркированного списка маркеры могут быть следующего вида:

| | |
|-------------|------|
| disc | круг |
|-------------|------|

| | |
|---------------|------------|
| circle | окружность |
|---------------|------------|

| | |
|---------------|---------|
| square | квадрат |
|---------------|---------|

| | |
|-------------|-------------|
| none | отсутствует |
|-------------|-------------|

либо, если мы хотим использовать свой рисунок маркера, то так:

```
list-style-image: url(images/bullet.gif);
```

Понятно, что картинка `bullet.gif` уже должна существовать в папке `images` вашего сайта.

Для нумерованных списков можно также задать различное отображение номеров:

| | |
|--------------------|---------------------------------|
| lower-roman | римские цифры в нижнем регистре |
|--------------------|---------------------------------|

| | |
|--------------------|------------------------------|
| upper-roman | то же, но в верхнем регистре |
|--------------------|------------------------------|

| | |
|-------------|-------------|
| none | отсутствует |
|-------------|-------------|