



Каскадные таблицы стилей - основа профессионального сайта

День 4. Часть 7

Кроссбраузерность CSS3

Кроссбраузерность — свойство сайта отображаться и работать во всех популярных браузерах идентично. Под идентичностью понимается отсутствие развалов верстки и способность отображать материал с одинаковой степенью читабельности. Понятие «кроссбраузерность» очень часто путают с попиксельным соответствием, что на самом деле является разными понятиями. Также **кроссбраузерность** – это "правильное" написание HTML и CSS, которое обеспечивают возможность отображения страниц сайта одинаково во всех браузерах. Требуются определенные действия для правильного (желательно валидного) отображения вашего контента. Кроссбраузерность как проблема уменьшается год от года. Если смотреть глобально, то путь к соблюдению стандарта можно сравнить с подъёмом на вершину пирамиды с разных её углов четырёх человек. В конце концов все четверо окажутся на вершине в одной точке. Аналогичная картина происходит сейчас с обозревателями - спорных моментов всё меньше. Соответственно, кроссбраузерности добиться проще, нежели несколько лет назад.



Кроссбраузерность CSS3

Спецификации HTML - это набор неких "усреднённых" правил, которые должны работать (являются рекомендациями) в браузерах.

Иными словами при верстке мы задаем рекомендации по отображению страницы для браузера. Делается это по той причине, что каждый браузер имеет свои "правила" по отображению страниц, которые придумывают разработчики этих самих браузеров. А так каждый браузер будет отображать примерно одинаково (чтобы не было примерно, а точно - это уже в руках верстальщика).

Делается это все с помощью тега DOCTYPE, который вставляется в начало HTML.

Стандарт HTML 4.01 Strict (строгий), Transitional (переходный), Frameset (с фреймами) соответственно:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">  
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

Кроссбраузерность CSS3

Стандарт XHTML 1.0 Strict (строгий), Transitional (переходный), Frameset (с фреймами)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">  
  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Стандарт XHTML 1.1 DTD

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Стандарт HTML 5

```
<!DOCTYPE html>
```

Поэтому выбирайте один из подходящих для вас вариантов и прописывайте в HTML-код на сайте. Указав спецификацию, намного проще и быстрее подогнать CSS под все браузеры.

Кроссбраузерность CSS3

Для решения проблем с кроссбраузерностью CSS3 есть несколько приёмов. Особой проблематичностью отличается браузер Internet Explorer. Большинство разработчиков CSS сегодня предпочитают использовать для обращения к IE условные комментарии.

Условные комментарии — это особая форма комментариев HTML, прочитать которые способен только IE. Это допустимый код HTML, и он не причиняет ущерба никаким браузерам — все остальные браузеры, кроме IE, пропускают условные комментарии точно так же, как любые другие комментарии HTML. Таким образом, вы можете добавлять код HTML, который будет прочитан только IE — всеми версиями или только определенными.

```
<!--[if IE]>
```

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

```
<![endif]-->
```

```
<!--[if IE 6]>
```

```
<link rel="stylesheet" href="ie_6.css" type="text/css">
```

```
<![endif]-->
```

```
<!--[if IE 7]>
```

```
<link rel="stylesheet" href="ie_7.css" type="text/css">
```

```
<![endif]-->
```

```
<!--[if IE 8]>
```

```
<link rel="stylesheet" href="ie_8.css" type="text/css">
```

```
<![endif]-->
```



Визуализационные эффекты на CSS3

В **CSS3** вы найдете множество новых свойств, позволяющих создавать визуальные эффекты, которые раньше требовали обязательного использования изображений (и иногда написания сценариев): скругленные углы, падающие тени, полупрозрачные фоны, градиенты и изображения в качестве рамок полей. Часть этих новых свойств принадлежит модулю **Backgrounds and Borders (Фон и границы)**; другие вы найдете в модулях **Colors (Цвета)** и **ImageValues (Значения изображения)**:

- свойство **word-wrap** не дает длинному тексту расползтись по странице;
- свойство **border-radius** предназначено для создания скругленных углов;
- цветовой синтаксис **RGBA** поможет создать полупрозрачные фоны;
- функция **linear-gradient** создает градиентные фоны;
- свойство **box-shadow** определяет падающие тени позади объектов;
- свойство **text-shadow** создает падающие тени позади текста;
- свойство **transform** позволяет поворачивать объекты.



Свойство word-wrap

В CSS3 наконец-то появился простой способ задать браузеру разрывать длинные строки текста, причем прямо внутри слов. Для этого нужно присвоить свойству **word-wrap** значение **break-word**, и браузер сам будет вставлять разрыв в подходящем месте строки, не давая ей вылезти за пределы блока.

Свойство `word-wrap` входит в модуль **Text (Текст)**. Оно указывает, разрешается ли браузеру разрывать строки прямо внутри слов. (Переносом строк с разрывами между слов управляет отдельное свойство `text-wrap`.)

Допустимые значения свойства `word-wrap`: `normal` (это значение по умолчанию) и `break-word`. Помимо переноса на новую строку длинных URL-адресов, свойство `word-wrap` удобно применять для форматирования данных в таблицах, для того чтобы ячейки не растягивались в ширину и не искажали макет страницы.

```
blockquote{
margin: 0 0 0 112px;
padding: 10px 15px 5px 15px;
border-top: 1px solid #fff;
background-color: #a6dadac;
word-wrap: break-word;
}
```

Свойство `border-radius`

Свойство `border-radius` входит в модуль **Backgrounds and Borders (Фон и границы)**. Оно объединяет четыре свойства, указывающих степень скругления каждого из четырех углов поля, в следующем порядке:

```
border-top-left-radius border-top-right-radius  
border-bottom-right-radius, border-bottom-left-radius
```

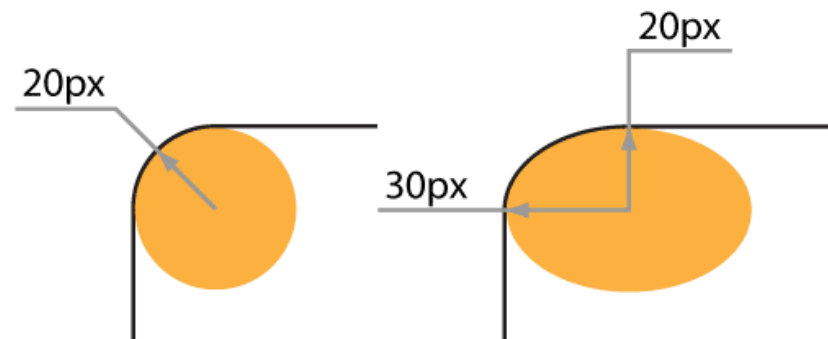
Все четыре значения можно перечислить внутри одного свойства `border-radius`, разделив их пробелами; можно также использовать одно общее значение для единообразного скругления всех углов. С использованием свойства `border-radius` можно оформлять следующие элементы:

- кнопки;
- заголовки вкладок;
- диалоговые окна;
- круглые эмблемы;
- столбцовые диаграммы;
- смайлики

Свойство border-radius

Число значений	Результат
1	Радиус указывается для всех четырех уголков.
2	Первое значение задает радиус верхнего левого и нижнего правого уголка, второе значение — верхнего правого и нижнего левого уголка.
3	Первое значение задает радиус для верхнего левого уголка, второе — одновременно для верхнего правого и нижнего левого, а третье — для нижнего правого уголка.
4	По очереди устанавливает радиус для верхнего левого, верхнего правого, нижнего правого и нижнего левого уголка.

В случае задания двух параметров через слэш, то первый задает радиус по горизонтали, а второй по вертикали (эллиптические уголки). На рис. 1 показана разница между обычным скругленным уголком и эллиптическим уголком.



Свойство border-radius

```
4 <style>
5   .radius {
6     background: #f0f0f0; /* Цвет фона */
7     border: 1px solid black; /* Параметры рамки */
8     padding: 15px; /* Поля вокруг текста */
9     margin-bottom: 10px; /* Отступ снизу */
10  }
11 </style>
12 <body>
13   <div style="border-radius: 50px 0 0 50px;" class="radius">
14     border-radius: 50px 0 0 50px;
15   </div>
16   <div style="border-radius: 40px 10px" class="radius">
17     border-radius: 40px 10px;
18   </div>
19   <div style="border-radius: 10em/1em;" class="radius">
20     border-radius: 13em/3em;
21   </div>
22   <div style="border-radius: 8px;" class="radius">
23     border-radius: 8px;
24   </div>
25 </body>
26 </html>
```



border-radius: 50px 0 0 50px;

border-radius: 40px 10px;

border-radius: 13em/3em;

border-radius: 8px;



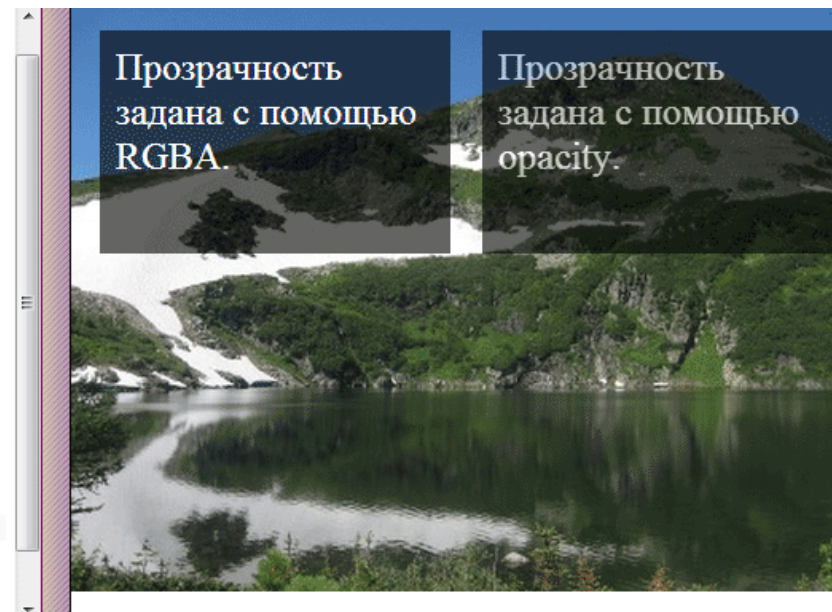
Цветовой синтаксис RGBA

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

RGBA добавлен в CSS3, поэтому валидацию CSS-кода надо проводить именно по этой версии. Следует отметить, что стандарт CSS3 ещё находится в разработке и некоторые возможности в нем могут поменяться.

Обратите внимание: задание прозрачности с помощью **RGBA** отличается от действия свойства `opacity` тем, что `opacity` делает прозрачным сам элемент и все его элементы потомки, а **RGBA** делает прозрачным только сам элемент.

```
2 <style type='text/css'>
3 #exbody {
4 background-image:url(mountaingl.jpg);
5 background-repeat:no-repeat;
6 background-size:500px 370px;}
7 #wrap1,#wrap2 {
8 width:200px;
9 height:120px;
10 margin:10px;
11 float:left;
12 padding:10px;
13 font-size:1.5em;
14 background-color:black;
15 color:white;
16 z-index:100;}
17 #wrap1 {
18 background-color:rgba(0,0,0,0.6);}
19 #wrap2 {
20 opacity:0.6;}
21 </style>
22 <body id="exbody">
23 <div id="wrap1">Прозрачность задана с помощью RGBA.</div>
24 <div id="wrap2">Прозрачность задана с помощью opacity.</div>
25 </body>
```



Функция **linear-gradient**

Обратите внимание: градиенты поддерживаются во всех современных браузерах, но требует добавления специального префикса. Для браузера IE10+ требуется префикс `-ms`, для Chrome и Safari префикс `-webkit`, для Opera префикс `-o` и для Firefox префикс `-moz`.

Так как в CSS3 сам браузер отрисовывает градиенты, то необходимость градиентных картинок отпадает и это позволяет увеличить скорость загрузки страниц.

Линейные градиенты в CSS3 создаются с помощью метода **linear-gradient**, который должен указываться в значение свойства **background**.

Для того чтобы создать линейный градиент необходимо указать его направление (может задаваться с помощью ключевых слов или градусов) и цвета перехода.

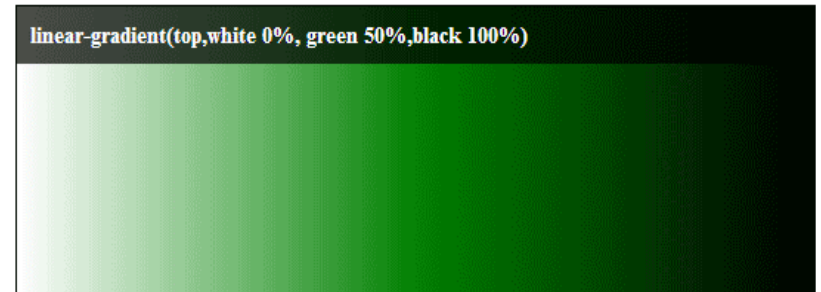
Цвета перехода - это цвета, которые принимает градиент в определенных его точках, например градиент, который плавно изменяет цвет с белого на черный имеет белый цвет перехода в начальной точке и черный в конечной.

Линейные градиенты могут иметь неограниченное количество цветов перехода.

Вы можете указывать координаты местоположения цветов с помощью % (0% подразумевает начало градиента, 100% конец).

Функция linear-gradient

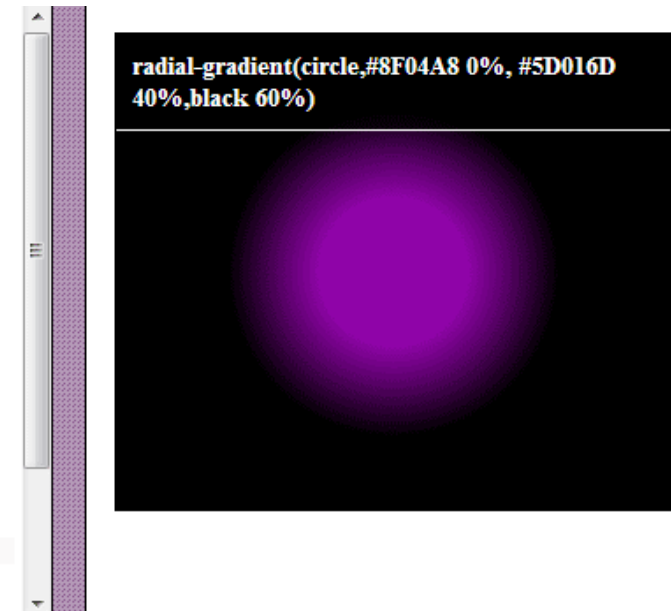
```
1 <!-- Блок wrap -->
2
3
4 #wrap1, #wrap2 {
5   margin:10px;
6   height:200px;
7   width:600px;
8   border:1px #000 solid;
9   float:left;}
10 #text1, #text2 {
11   background-color:rgba(0,0,0,0.7);
12   color:white;
13   padding:10px;
14   font-weight:bold;}
15 #wrap1 {
16   background:linear-gradient(left,white 0%,green 50%,black 100%);
17   background:-webkit-linear-gradient(left,white 0%,green 50%,black 100%); /* для Chrome и Safari */
18   background:-o-linear-gradient(left,white 0%,green 50%,black 100%); /* для Opera */
19   background:-moz-linear-gradient(left,white 0%,green 50%,black 100%); /* для Firefox */
20   background:-ms-linear-gradient(left,white 0%,green 50%,black 100%); /* для IE10+ */}
21 </style>
22 </head>
23 <body>
24 <div id="wrap1"> <div id="text1">linear-gradient(top,white 0%, green 50%,black 100%)</div> </div>
25 </body>
26 </html>
```



Функция radial-gradient

С помощью метода **radial-gradient** можно создавать сферические градиенты. Синтаксис определения сферических градиентов очень похож на синтаксис линейных, но требует также задания формы градиента (может быть сферической или эллипсоидной).

```
1 <html>
2 <head>
3 <style type="text/css">
4 #wrap2 {
5 margin:10px;
6 height:300px;
7 width:350px;
8 border:1px #000 solid;
9 float:left;}
10 #text2 {
11 background-color:rgba(0,0,0,0.7);
12 color:white;
13 padding:10px;
14 font-weight:bold;
15 border-bottom:1px white solid;}
16 #wrap2 {
17 background:radial-gradient(circle,#8F04A8 0%,#5D016D 40%,black 60%);
18 background:-webkit-radial-gradient(circle,#8F04A8 20%,#5D016D 30%,black 45%); /* для Chrome и Safari */
19 background:-o-radial-gradient(circle,#8F04A8 0%,#5D016D 40%,black 60%); /* для Опера */
20 background:-moz-radial-gradient(circle,#8F04A8 0%,#5D016D 40%,black 60%); /* для Firefox */
21 background:-ms-radial-gradient(circle,#8F04A8 0%,#5D016D 40%,black 60%); /* для IE10+ */
22 </style>
23 </head>
```



Свойство box-shadow

Свойство **box-shadow** дает возможность добавлять к блочным элементам множественные тени (внутренние и внешние). Для этого тебе нужно указать значения цвета, размера, размытости и смещения (color, size, blur и offset).



```
1 <html>
2 <head>
3 <meta charset="utf-8">
4 <title>box-shadow</title>
5 <style>
6   .shadow {
7     background: #fc0; /* Цвет фона */
8     -moz-box-shadow: 0 0 10px rgba(0,0,0,0.5); /* Для Firefox */
9     -webkit-box-shadow: 0 0 10px rgba(0,0,0,0.5); /* Для Safari и Chrome */
10    box-shadow: 0 0 10px rgba(0,0,0,0.5); /* Параметры тени */
11    padding: 10px;
12  }
13 </style>
14 </head>
15 <body>
16 <div class="shadow">В чашах юга жил бы цитрус? Да, но фальшивый экземпляр!</div>
17 </body>
18 </html>
```

В чашах юга жил бы цитрус? Да, но фальшивый экземпляр!

Свойство text-shadow

Свойство **text-shadow** задает эффект тени тексту элемента. При этом размер элемента не меняется (несмотря на то что тень может быть расположена далеко за пределами элемента).

Значения свойства задают смещение тени, ее радиус и цвет:

[цвет] [смещение тени по оси X] [смещение тени по оси Y] [радиус тени]

или

[смещение тени по оси X] [смещение тени по оси Y] [радиус тени] [цвет].

Порядок значений строго определен. Смещение тени задается относительно верхнего левого угла текста. Положительные значения смещения сдвигают тень вправо и вниз, отрицательные влево и вверх.

отрицательные значения смещений

Текст

положительные значения смещений тени

Текст

```
1 <html>
2 <head>
3 <title></title>
4 <style type="text/css">
5 h1 {
6     text-shadow: 0px 3px 3px #000;
7 }
8 </style>
9 </head>
10 <body>
11 <h1>Заголовок с тенью</h1>
12 </body>
13 </html>
```

Заголовок с тенью

Свойство transform

Свойство **transform** — применяет трансформацию к элементу (можно применить несколько трансформаций, задавая значения через пробел).

Элемент можно передвигать, масштабировать, поворачивать и наклонять.

Функция	Описание
translate(x,y)	Смещает элемент от изначальной позиции по горизонтали и вертикали.
translateX(x)	Смещает элемент по горизонтали.
translateY(y)	Смещает элемент по вертикали.
scale(x,y)	Растягивает элемент по вертикали и горизонтали.
scaleX(x)	Растягивает элемент по горизонтали.
scaleY(y)	Растягивает элемент по вертикали.
rotate(градусы)	Поворачивает элемент по часовой стрелке.
skew(x,y)	Скашивает элемент по горизонтали и вертикали.
skewX(x)	Скашивает элемент по горизонтали.
skewY(y)	Скашивает элемент по вертикали.
matrix(x,x,x,x,x,x)	Трансформирует элемент с помощью матрицы из 6 значений.

Свойство transform: translate

Translate сдвигает элемент на заданное значение по горизонтали и вертикали.

transform: translate(tx,ty)

```
1 <html>
2 <head>
3 <style type='text/css'>
4 #wrap1,#el1,#el2,#el3,#el4 {
5 width:220px;
6 height:120px;
7 border:1px #000 solid;
8 padding:10px;}
9 #el1 {
10 transform:translate(100px,30px);
11 -o-transform:translate(100px,30px);
12 -webkit-transform:translate(100px,30px);
13 -moz-transform:translate(100px,30px);
14 -ms-transform:translate(100px,30px);
15 background-color:#f4e4ec;}
16 </style></head>
17 <body>
18 <p><b>1. Демонстрация функции translate(): </b></p>
19 <div id="wrap1">
20 <div id="el1">Я смещен на 100 пикселей по горизонтали и 50 пикселей по вертикали от моей первонач
21 </div>
22 <br /><br /></div></body></html>
```

1. Демонстрация функции translate():

Я смещен на 100 пикселей по горизонтали и 50 пикселей по вертикали от моей первоначальной позиции

Свойство transform:scale

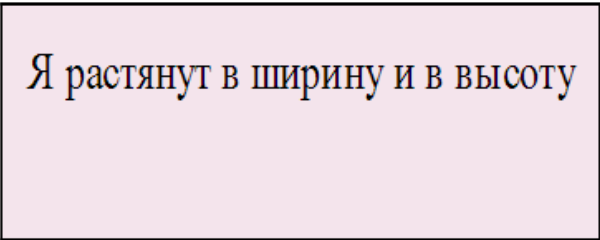
Scale - масштаб элемента по горизонтали и вертикали.

```
transform: scale(sx, sy);
```

Значение больше 1 увеличивает масштаб элемента, меньше 1 — уменьшает масштаб.

```
1 <html>
2 <head>
3 <style type='text/css'>
4 #wrap1, #el1, #el2, #el3, #el4 {
5 width:220px;
6 height:120px;
7 border:1px #000 solid;
8 padding:10px;}
9 #el3 {
10 transform:scale(1.5,2);
11 -o-transform:scale(1.5,2);
12 -webkit-transform:scale(1.5,2);
13 -moz-transform:scale(1.5,2);
14 -ms-transform:scale(1.5,2);
15 background-color:#f4e4ec;
16 margin-left:100px;
17 margin-top:70px;
18 height:50px;}
19 </style>
20 </head>
21 <body>
22 <p><b>3. Демонстрация функции scale(): </b></p>
23 <div id="el3">Я растянут в ширину и в высоту</div>
24 <br /><br /></body></html>
```

3. Демонстрация функции scale():



Я растянут в ширину и в высоту

Свойство transform: rotate

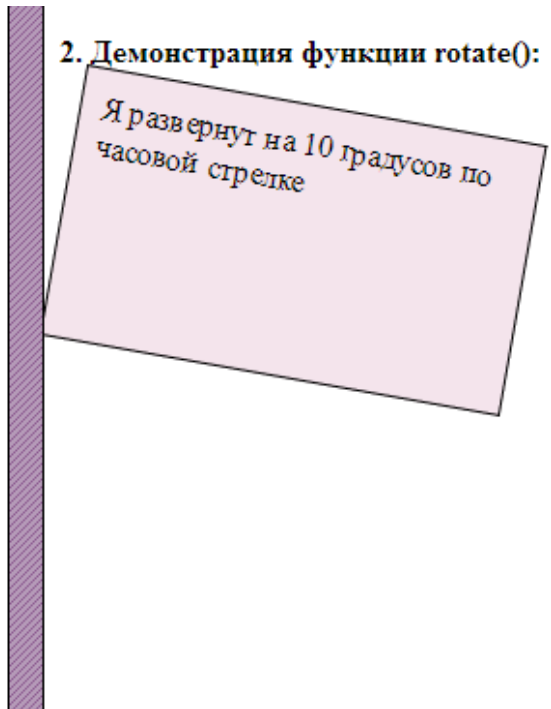
Rotate - поворот элемента на заданный угол относительно точки трансформации, задаваемой свойством transform-origin.

transform: rotate(<угол>) ;

```
1 <html>
2 <head>
3 <style type='text/css'>
4 #wrap1,#el1,#el2,#el3,#el4 {
5 width:220px;
6 height:120px;
7 border:1px #000 solid;
8 padding:10px;
9 }
10 #el2 {
11 transform:rotate(10deg);
12 -o-transform:rotate(10deg);
13 -webkit-transform:rotate(10deg);
14 -moz-transform:rotate(10deg);
15 -ms-transform:rotate(10deg);
16 background-color:#f4e4ec;
17 }
18 </style>
19 </head>
20 <body>
21 <p><b>2. Демонстрация функции rotate(): </b></p>
22 <div id="el2">Я развернут на 10 градусов по часовой стрелке</div></body></html>
```

2. Демонстрация функции rotate():

Я развернут на 10 градусов по
часовой стрелке

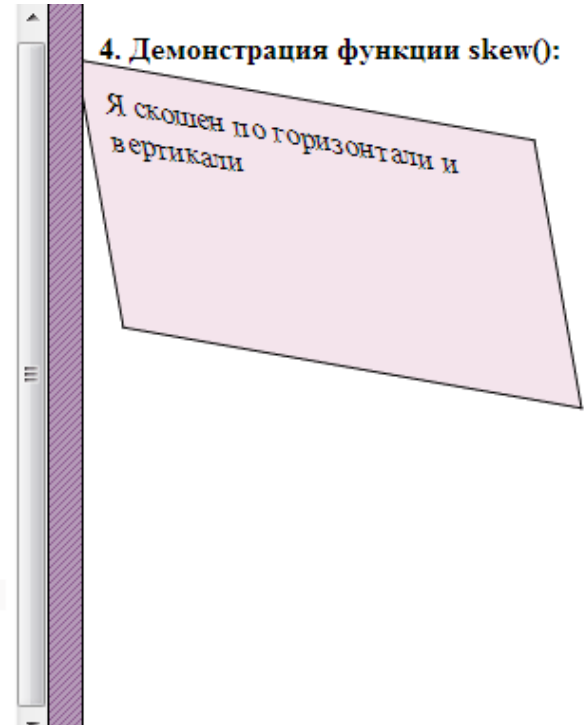


СВОЙСТВО transform:scew

Scew - скашивает элемент по горизонтали и вертикали.

transform: scew (<угол>, <угол>);

```
1 <html>
2 <head>
3 <style type='text/css'>
4 #wrap1, #el1, #el2, #el3, #el4 {
5 width:220px;
6 height:120px;
7 border:1px #000 solid;
8 padding:10px;
9 }
10 #el4 {
11 transform:skew(10deg,10deg);
12 -o-transform:skew(10deg,10deg);
13 -webkit-transform:skew(10deg,10deg);
14 -moz-transform:skew(10deg,10deg);
15 -ms-transform:skew(10deg,10deg);
16 background-color:#f4e4ec;}
17 </style>
18 </head>
19 <body>
20 <p><b>4. Демонстрация функции skew(): </b></p>
21 <div id="el4">Я скошен по горизонтали и вертикали</div>
22 </body>
23 </html>
24
```



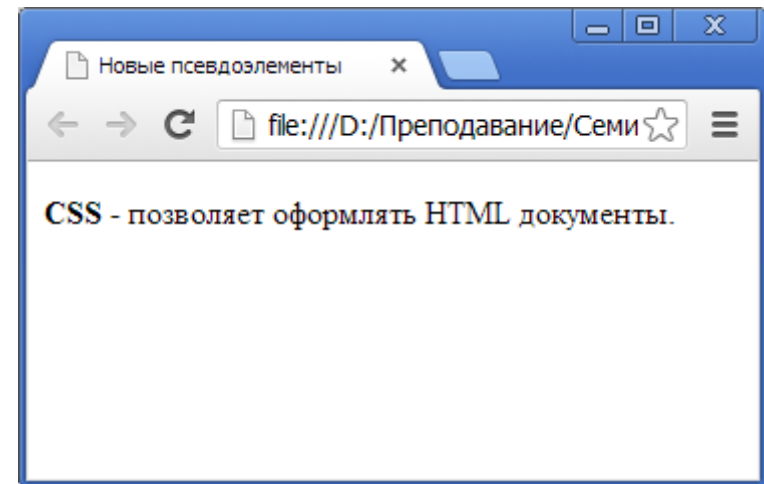
Псевдоэлементы :before :after

Псевдоэлементы `:before` и `:after` используется для вставки автоматически генерируемого контента перед элементом и после элемента соответственно. При этом генерируемым может быть любой элемент (кроме форм), например картинка. Свойство применимо ко всем элементам. Данные псевдоэлементы работают совместно со свойством `content`.

Содержание веб-страницы

```
<!DOCTYPE html>
<head>
<title>Новые псевдоэлементы</title>
<style type="text/css">
  p:before{
    content:"CSS - ";
    font-weight:bold;
  }
  p:after{
    content:" HTML документы.";
  }
</style>
</head>
<body>
<p>позволяет оформлять</p>
</body>
</html>
```

Отображение веб-страницы

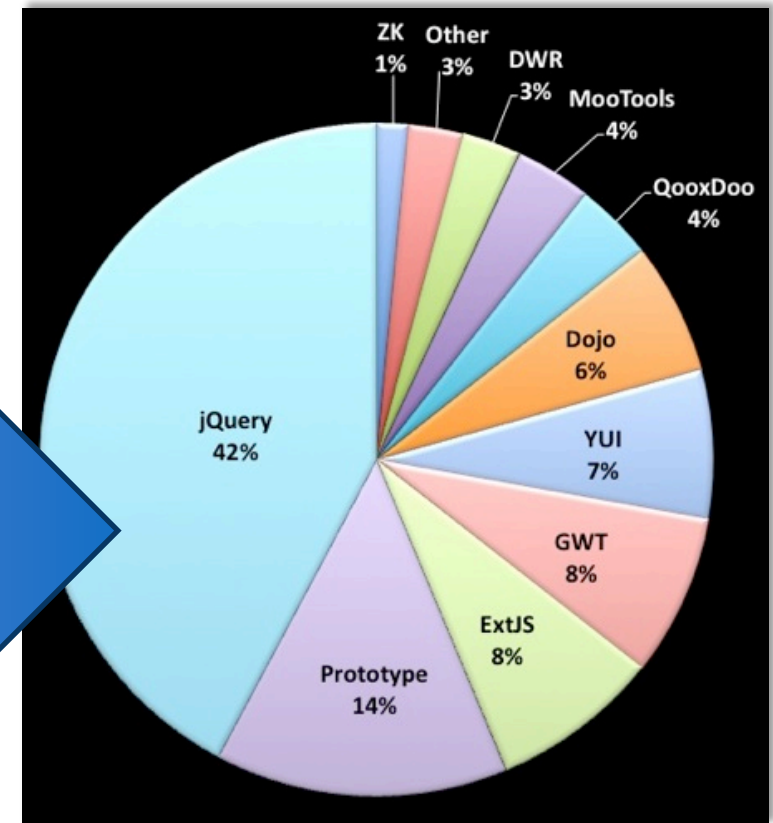


Совместное использование JavaScript и CSS

JavaScript – объектно-ориентированный скриптовый язык, который уже долгое время является признанным стандартом для программирования на стороне клиентского Web-браузера. В настоящее время сложно представить себе современные интерактивные сайты без этого скриптового языка. Для упрощения и ускорения разработки больших сайтов стали массово появляться целые платформы на JavaScript, которые представляли собой уже готовые библиотеки с набором стандартных возможностей:

Рассмотрим несколько ведущих фреймворков по сумме 4 критериев:

- ✓ наиболее широкой распространенности и популярности;
- ✓ качественности сопутствующей документации;
- ✓ показателям производительности;
- ✓ легкости освоения, логичности и простоты интеграции в сайт.



Совместное использование JavaScript и CSS

1 место: jQuery. Размер: 155Kb (миним.версия – 72Kb), последняя версия: 1.9 (16 января 2013) **jQuery** – самая популярная и распространенная библиотека на JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML.

Другой секрет популярности этой библиотеки, это реализация очень выдержанного баланса между хорошей функциональностью и при этом удачной попыткой создания достаточно компактного универсального ядра. Если дополнить сюда компромиссную возможность очень просто расширять функциональность базовой библиотеки за счет включения дополнительных плагинов (количество которых сейчас зашкаливает уже за 1000) – то на выходе мы получаем по-настоящему очень универсальный, гибкий и быстрый продукт.

Для желающих получить более продвинутые UI-эффекты (UI, userinterface), такие как масштабирование, сортировка, поддержка draganddrop, и стандартные виджеты, такие как закладки, слайдер, прогрессбар и т.д., — рекомендуется использовать родственную библиотеку jQuery UI, которая специализируется именно на графических эффектах и созданию сложного пользовательского интерфейса.



Совместное использование CSS и jQuery

jQuery - совокупность javascript функций, упрощающих написание скриптов на javascript. Для того чтобы использовать jQuery необходимо подключить файл js(javascript), как правило в блок `<head>...</head>`:

```
<script type="text/javascript" src="jquery-1.4.4.min.js"></script>
```

Возможности jQuery в CSS

Прежде всего jQuery - это быстрый и легкий доступ к CSS свойствам и их модификации. Вам не составит труда написать:

```
1.$ ("div").css ("color", "blue");
```

и изменить текст всех элементов DIV на голубой. Или

```
1.$ ("div").css ("width", "500px");
```

и тем самым изменить ширину всех элементов DIV, которая теперь будет соответствовать 500px.

Огромное значение в CSS имеют селекторы, шаблоны селекторов. В jQuery любые селекторы поддерживаемые стандартом CSS3, реализованы кроссбраузерно, то есть те, которые не поддерживаются отдельными браузерами без использования jQuery, полностью поддерживаются с использованием jQuery.

Методы jQuery для CSS свойств

Основными методами CSS следует считать:

- `css("property", "value")`
- `css({"property": "value", "property": "value", ...n})`
- `addClass("class")`
- `removeClass("class")`
- `hasClass("class")`
- `toggleClass("class")`

В качестве примера рассмотрим метод переключения CSS-класса `toggleClass("class")`. Пример:

```
1.$("DIV.news").toggleClass("hot");
```

приведет к тому, что к элементу с классом добавится дополнительный класс "hot".

```
1.<div class="news">Текст новости.</div>
```

то есть предшествующий HTML будет выглядеть следующим образом:

```
1.<div class="newshot">Текст новости.</div>
```