

Оглавление

Предисловие	12
Глава 1. Свойства и значения CSS	13
1.1. Внешнее размещение	14
1.2. Внутреннее размещение.....	15
1.3. Строковое размещение	15
1.4. Селекторы	16
1.5. Взаимосвязь между свойствами и значениями	21
1.6. Комментарии в CSS-коде	22
1.7. Оформление стилей	23
1.8. CSS-переменные	25
1.9. Метаязык SASS.....	26
1.10. Суть каскадных таблиц стилей.....	27
1.11. Селекторы CSS.....	30
1.12. Лояльность CSS.....	30
1.13. Распространенные комбинации	30
1.14. Сокращенные нотации	31
Глава 2. Псевдоэлементы	32
2.1. ::after	32
2.2. ::before	32
2.3. ::first-letter	32

2.4.	::first-line.....	33
2.5.	::selection	33
2.6.	::slotted(*).....	33
Глава 3.	Псевдоселекторы.....	34
3.1.	:link.....	36
3.2.	:visited	36
3.3.	:hover	36
3.4.	:active.....	37
3.5.	:focus	37
3.6.	:enabled	37
3.7.	:disabled	37
3.8.	:default.....	38
3.9.	:indeterminate	38
3.10.	:required.....	38
3.11.	:optional	38
3.12.	:read-only.....	39
3.13.	:root	39
3.14.	:only-of-type.....	39
3.15.	:first-of-type	39
3.16.	:nth-of-type().....	40
3.17.	:last-of-type	40
3.18.	:nth-child().....	40
3.19.	:nth-last-child()	41
3.20.	:nth-child(odd)	41
3.21.	:nth-child(even).....	41
3.22.	:not().....	42
3.23.	:empty	42
3.24.	Вложенные псевдоселекторы.....	42
3.25.	:dir rtl) и :dir(ltr).....	42
3.26.	:only-child.....	43

Глава 4. Блочная модель CSS	44
Глава 5. Позиционирование.....	48
5.1. Тестовый элемент	48
5.2. Статичное и относительное позиционирование	49
5.3. Абсолютное и фиксированное позиционирование	51
5.4. Фиксированное позиционирование	54
5.5. «Липкое» позиционирование.....	55
Глава 6. Работа с текстом	56
6.1. Свойство text-align.....	59
6.2. Свойство text-align-last	60
6.3. Свойство overflow.....	61
6.4. Свойство text-decoration-skip-ink.....	63
6.5. Свойство text-rendering.....	63
6.6. Свойство text-indent.....	64
6.7. Свойство text-orientation.....	64
6.8. Свойство text-shadow.....	67
Глава 7. Свойства margin, border-radius, box-shadow и z-index	70
Глава 8. Логотип Nike.....	76
Глава 9. Свойство display	79
Глава 10. Свойство visibility.....	82
Глава 11. Плавающие элементы	83
Глава 12. Цветовые градиенты.....	84
12.1. Общие сведения.....	84
12.2. Типы градиентов.....	88

Глава 13. Фильтры	93
13.1. Фильтр blur().....	93
13.2. Фильтр brightness()	94
13.3. Фильтр contrast()	94
13.4. Фильтр grayscale()	94
13.5. Фильтр opacity().....	94
13.6. Фильтр hue-rotate().....	95
13.7. Фильтр invert()	95
13.8. Фильтр saturate()	95
13.9. Фильтр sepia()	95
13.10. Фильтр drop-shadow()	95
Глава 14. Фоновые изображения	96
14.1. Указание нескольких значений	101
14.2. Свойство background-position.....	101
14.3. Фон из нескольких изображений	103
14.4. Прозрачность фона.....	104
14.5. Множественные фоны.....	104
14.6. Свойство background-attachment.....	107
14.7. Свойство background-origin.....	108
Глава 15. Свойство object-fit	110
Глава 16. Границы	111
16.1. Эллиптический радиус границы.....	114
Глава 17. 2D-трансформации	117
17.1. Свойство translate	117
17.2. Свойство rotate.....	118
17.3. Свойство transform-origin.....	120
Глава 18. 3D-трансформации	122
18.1. Свойство rotateX	122
18.2. Свойства rotateY и rotateZ.....	123
18.3. Свойство scale.....	123

18.4. Свойство translate	124
18.5. Создание 3D-куба.....	125
Глава 19. Flex-верстка	127
19.1. Свойство display: flex.....	127
19.2. Главная и перекрестная оси.....	128
19.3. Свойство flex-direction	129
19.4. Свойство flex-wrap	129
19.5. Свойство flex-flow	130
19.6. Свойство justify-content.....	132
19.7. Свойство flex-basis	134
19.8. Свойство align-items.....	134
19.9. Свойство flex-grow	135
19.10. Свойство order.....	136
19.11. Свойство flex-shrink.....	137
19.12. Свойство justify-items	137
19.13. Интерактивный flex-редактор.....	138
Глава 20. Grid-верстка: блочная модель	139
Глава 21. Grid-верстка: шаблоны grid-областей	141
21.1. Grid-верстка и медиазапросы.....	142
21.2. Верстка сайта на основе grid-элементов.....	144
21.3. Неявные строки и столбцы	148
21.4. Свойство grid-auto-rows.....	149
21.5. Ячейки с автоматически изменяемой шириной.....	151
21.6. Промежутки	152
21.7. Единицы fr для эффективного определения размера оставшегося пространства.....	157
21.8. Работа с единицами fr	159
Глава 22. Единицы fr и промежутки	162
22.1. Повторение значений.....	165
22.2. Группировка	166

22.3.	Свойства grid-row-start и grid-row-end.....	169
22.4.	Краткая нотация диапазона.....	172
22.5.	Выравнивание контента в grid-элементах.....	175
22.6.	Свойство align-self.....	175
22.7.	Свойство justify-self.....	177
22.8.	Шаблоны grid-областей.....	178
22.9.	Наименование линий сетки.....	179
Глава 23.	Анимация.....	182
23.1.	Свойство animation.....	184
23.2.	Свойство animation-name.....	184
23.3.	Свойство animation-duration.....	185
23.4.	Свойство animation-delay.....	185
23.5.	Свойство animation-direction.....	186
23.6.	Свойство animation-iteration-count.....	187
23.7.	Свойство animation-timing-function.....	188
23.8.	Свойство animation-fill-mode.....	191
23.9.	Свойство animation-play-state.....	191
Глава 24.	Прямая и обратная кинематика.....	192
Глава 25.	Принципы SASS/SCSS.....	194
25.1.	Новый синтаксис.....	194
25.2.	Необходимые условия.....	195
25.3.	Расширенные возможности.....	196
25.4.	Переменные.....	196
25.5.	Вложенные правила.....	197
25.6.	Директива &.....	198
25.7.	Примеси.....	198
25.8.	Поддержка разных браузеров.....	199
25.9.	Арифметические операторы.....	200
25.10.	Операторы управления потоком.....	205

25.11. Функции SASS.....	209
25.12. Тригонометрические функции SASS.....	210
25.13. Пользовательские функции SASS.....	210
25.14. Анимация генератора.....	212
Глава 26. CSS-графика: Tesla	215
Послесловие	222
Благодарности	222
От издательства.....	223

14 Фоновые изображения

Так вы думаете, что знаете HTML-фоны? Ну, может, знаете, а может, и нет. Данная глава, надеюсь, познакомит читателя с общей картиной. Мы рассмотрим несколько свойств CSS, способных помочь изменить настройки фонового изображения для любого элемента HTML.

Пример использования свойства `background: url("image.jpg")` или `background-image: url("image.jpg")`:



Изображение, используемое в этой главе, — очаровательный котенок на полосатом фоне.

Если размеры элемента превышают размеры исходного изображения, то оно будет повторяться внутри тела данного элемента, повторяя

заполнение оставшихся сторон элемента содержимым изображения. Это почти как растягивание конечных обоев по всей области элемента.



Чтобы установить фоновое изображение для любого элемента, можно использовать следующие команды CSS:

```
001 background: url("kitten.jpg");
```

или...

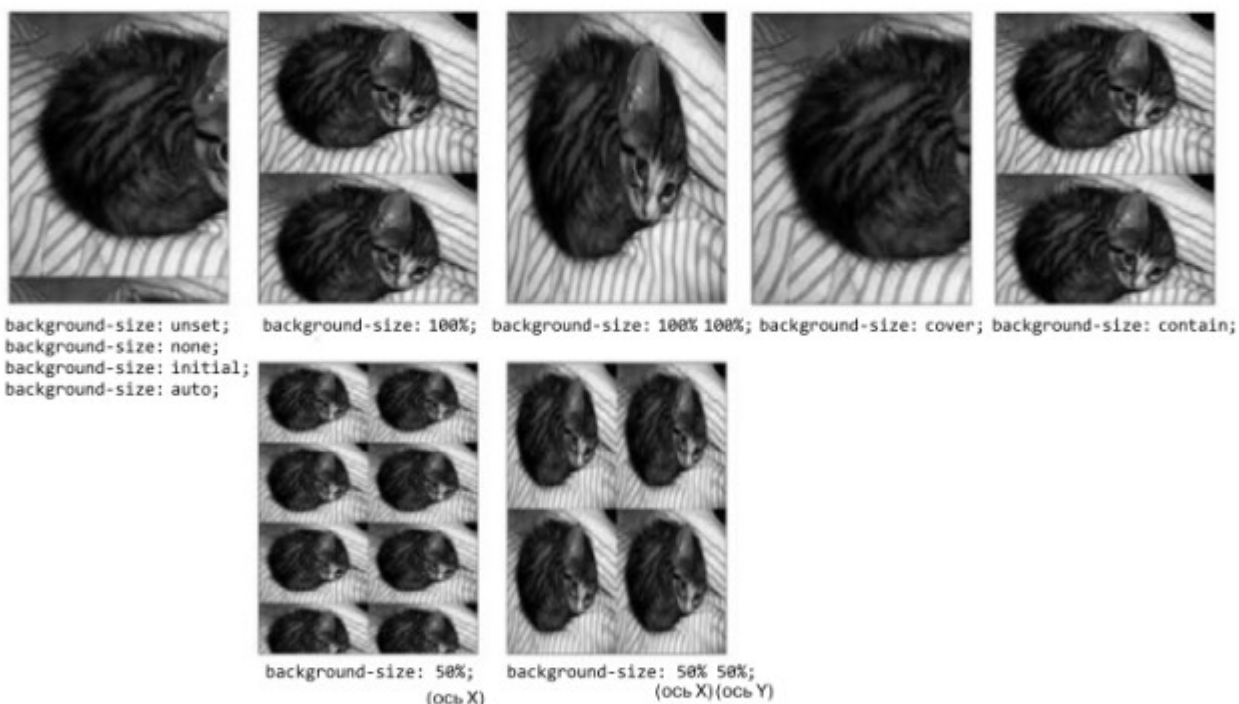
```
001 background-image: url("kitten.jpg");
```

Кроме того, можно задействовать внутренний CSS-код, поместив его между тегами `<style></style>`.

Посмотрим на тот же фон котенка... только с установленным значением `background-repeat: no-repeat`:



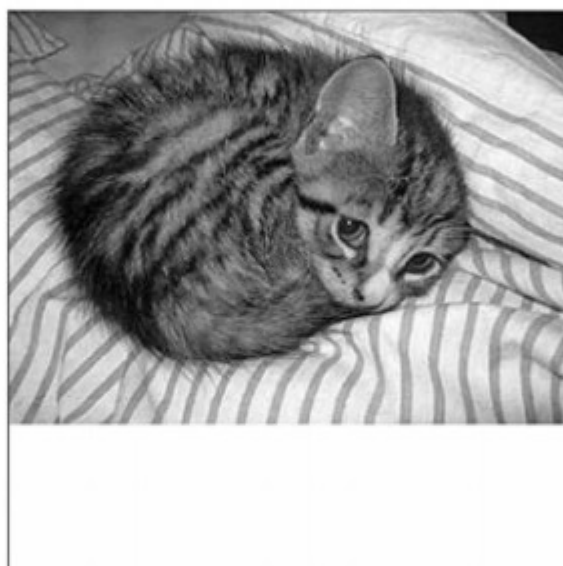
Далее показаны результаты, созданные с помощью свойства `background-size`. Слева направо приведены следующие примеры: (`unset` — `none` — `initial` — `auto`), которые приводят к значению по умолчанию.



Значение `100%` растянет изображения по горизонтали, но не по вертикали. Значение `100% 100%` растянет изображение по всему доступному пространству. Значение `cover` растянет изображение по всему вертикальному пространству элемента, обрежет все в горизонтальном

направлении, как при переполнении. Значение `contain` гарантирует, что изображение растягивается по горизонтали по ширине элемента, и, сохраняя исходную пропорцию, растягивает его по вертикали столько, сколько потребуется, повторяя изображение до тех пор, пока оно не заполнит нижнюю часть элемента.

Объединяя свойства `background-repeat: no-repeat` и `background-size: 100%`, можно растянуть изображение только по горизонтали, по всей ширине элемента:



Что, если вы хотите повторить фон по вертикали, но держать его растянутым по ширине? Все возможно, просто удалите значение `no-repeat` из предыдущего примера. В конечном итоге получится вот что:



Как было упомянуто выше, такой фоновый метод HTML/CSS используется для сайтов, чье содержимое растягивается вертикально на большую область пространства. Я думаю, что одна из итераций сайта Blizzard задействовала его ранее. Иногда вы захотите обрезать содержимое и сделать статичным. В других случаях пожелаете, чтобы оно продолжалось все время по вертикали. Все будет зависеть от вашего видения макета.

Бывает необходимо растянуть изображение поперек, чтобы установить ограничивающую рамку элемента. Однако это зачастую приводит к некоторым искажениям. CSS автоматически растянет изображение в соответствии с некоторым автоматически рассчитанным процентным значением.



Нет необходимости говорить, что такой эффект будет наблюдаться только в том случае, если HTML-элемент и размер изображения не совпадают.

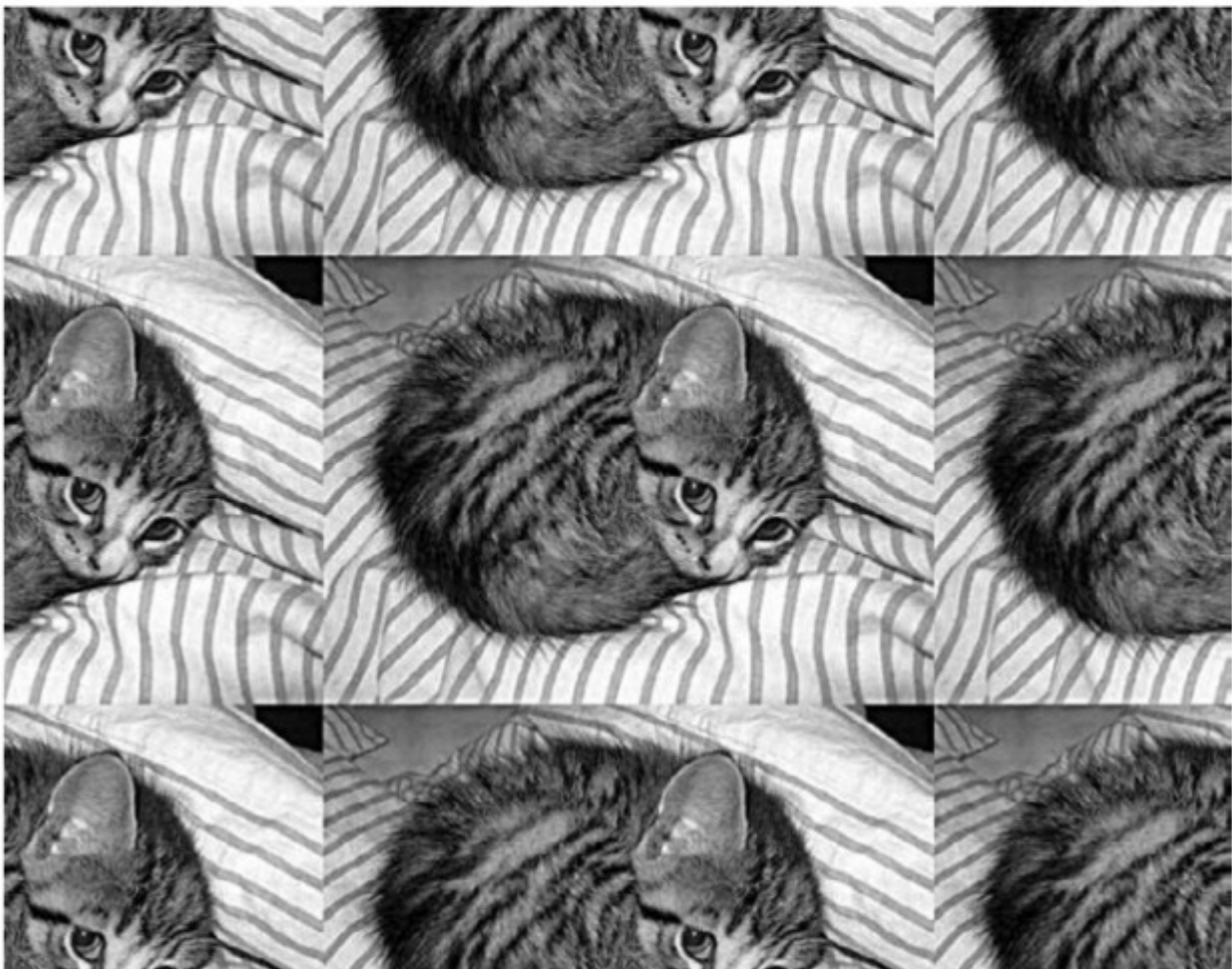
Как упоминалось выше: чтобы растянуть изображение, следует установить свойство `background-size: 100% 100%`. Обратите внимание: значение `100%` повторяется дважды. Первое значение растягивает изображение по вертикали, а второе делает то же самое по горизонтали. Вы можете использовать значения в диапазоне от `0` до `100%`, хотя я не вижу в этом необходимости.

14.1. Указание нескольких значений

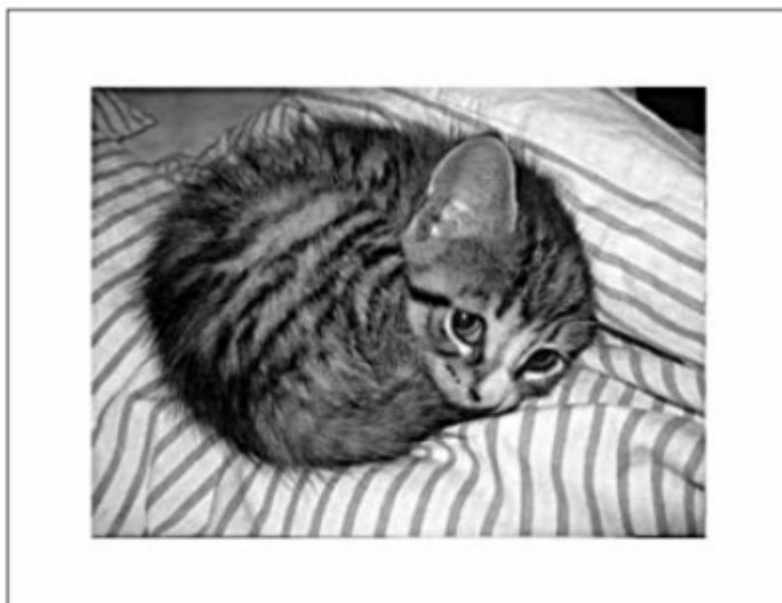
Всякий раз, когда вам нужно указать в HTML несколько значений, они часто разделяются пробелом. Вертикальные координаты (*ось Y*, или *высота*) всегда указываются первыми. Иногда значения разделяются запятой. Например, при необходимости указать несколько фонов они обычно разделяются запятой, а не пробелом. (В последнем разделе я покажу, как это делается.)

14.2. Свойство `background-position`

Далее приведен пример `background-position: center center`:



Вы можете центрировать изображение, но потерять повторяемость шаблона, указав значение `no-repeat` для свойства `background-repeat`.



Центрирование изображения:

```
001 background-position: center center;
```

Отключение повторения:

```
001 background-repeat: no-repeat;
```

Вы можете повторить изображение только по оси *X* (по горизонтали), используя значение `repeat-x`:



Вы можете легко отцентрировать и повторить изображение только по горизонтали, указав `repeat-x` в качестве значения свойства `background-repeat`.

Для того же эффекта, но по оси Y служит свойство значение repeat-y:



Как и любым другим свойством CSS, вы должны манипулировать значениями для достижения желаемых результатов. Я думаю, мы рассмотрели почти все, касающееся фонов. За исключением одной последней темы...

14.3. Фон из нескольких изображений

Можно добавить более одного фона к одному и тому же HTML-элементу. Процесс довольно прост.

Рассмотрим изображения, хранящиеся в двух отдельных файлах.

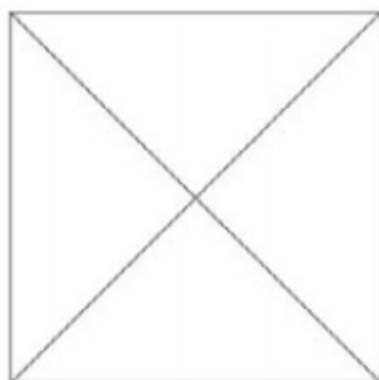


image1.png

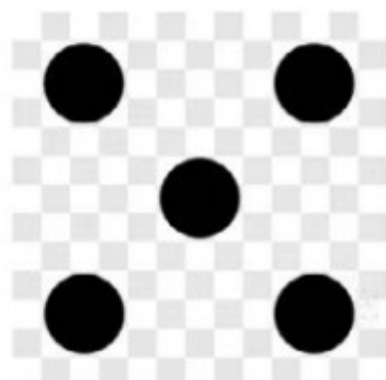


image2.png

Инструмент Magic Eraser (Волшебный ластик) в программе Photoshop можно выбрать на панели инструментов:

Рисунок в виде шахматной доски справа на рисунке используется только для обозначения прозрачности. Белые и сероватые квадраты не являются реальной частью самого изображения. Это та самая прозрачная область, которая обычно видна в программном обеспечении для цифровых манипуляций.



Когда изображение справа помещается поверх других HTML-элементов или изображений, клетчатая область не будет блокировать данный контент внизу. И в этом заключается вся идея множественных фонов в HTML.

14.4. Прозрачность фона

Чтобы полностью использовать преимущества множественных фонов, одно из фоновых изображений должно иметь прозрачную область. Как это сделать? В нашем примере второе изображение, `image2.png`, содержит пять черных точек на прозрачном фоне, обозначенном клетчатым узором.

Как и многие другие свойства CSS, которые принимают несколько значений для настройки множественных фонов, они предоставляют для свойства фона набор значений, разделенных запятой.

14.5. Множественные фоны

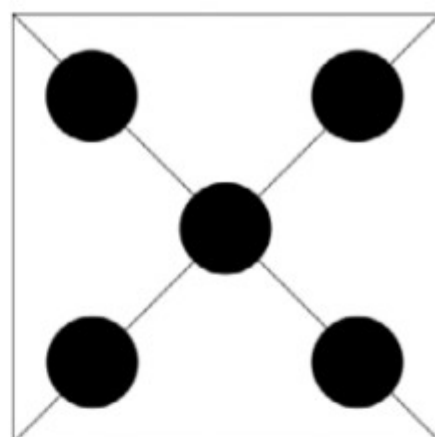
Чтобы назначить несколько *разделенных на слои* фоновых изображений, для одного и того же HTML-элемента можно использовать следующий код CSS:

```
001 body { background: url("image2.png"), url("image1.png"); }
```

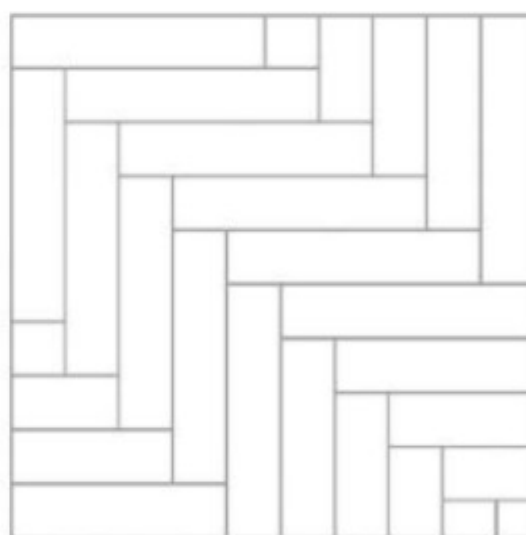

Очень важен порядок, в котором вы присваиваете изображения для свойства `url` фона. Обратите внимание: самое верхнее изображение всегда указывается первым. Вот почему мы начинаем с `image2.png`.

В данном примере мы рассмотрели множественные фоны в теории на субъективном элементе `div` (или аналогичном) в виде квадрата.

Возьмем другой пример.



puppy.png



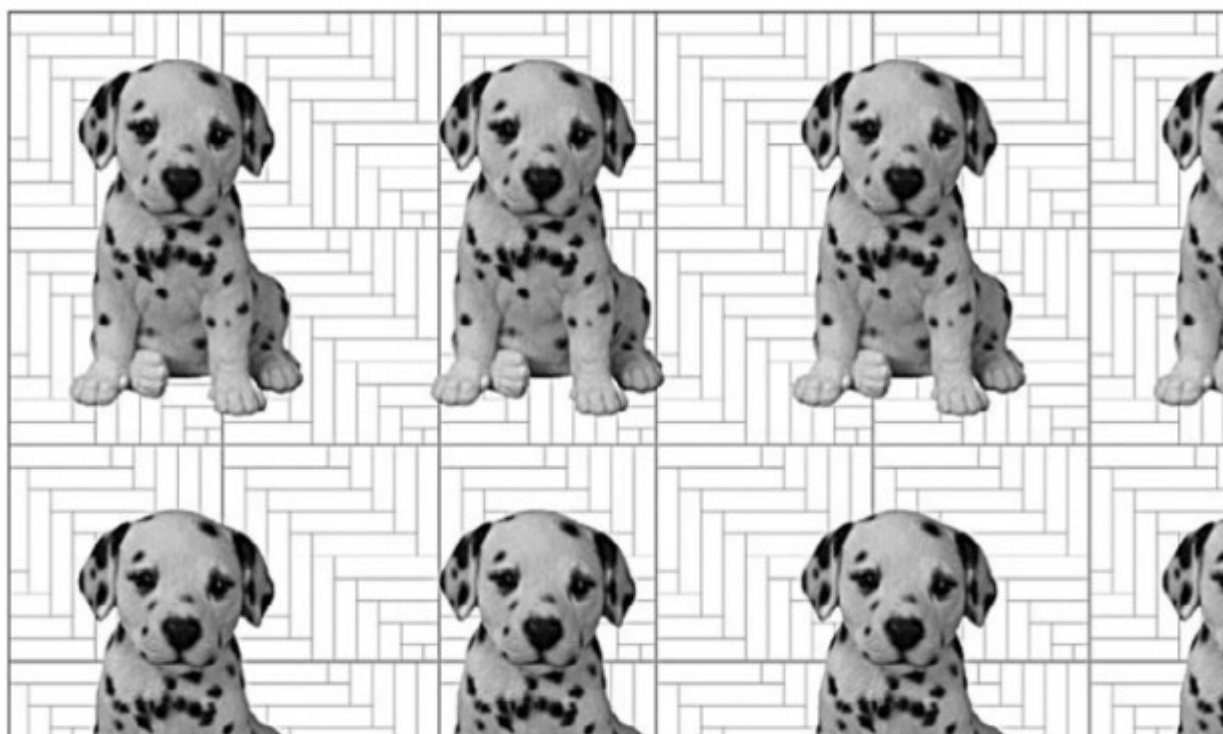
pattern.png

Обратите внимание: изображение `puppy.png` будет первым элементом в списке, разделенном запятыми. Это изображение, которое мы хотим наложить поверх всех других изображений в списке.

Объединяем два изображения:

```
001 body { background: url('puppy.png'), url('pattern.png')}
```

Получаем следующий результат.



Существуют и другие свойства фона, которые также принимают списки, разделенные запятыми. Это почти все остальные свойства, связанные с фоном, кроме `background-color`.

Таким же образом можно присвоить иные параметры для каждого отдельного фона с помощью других свойств фона, показанных ниже:

```
001 background  
002 background-attachment  
003 background-clip  
004 background-image  
005 background-origin  
006 background-position  
007 background-repeat  
008 background-size
```

Следующее свойство нельзя использовать со списком по очевидным причинам:

```
001 background-color
```

Что бы это значило — обеспечить несколько значений цвета для фона? Всякий раз, когда задается свойство `background-color`, оно

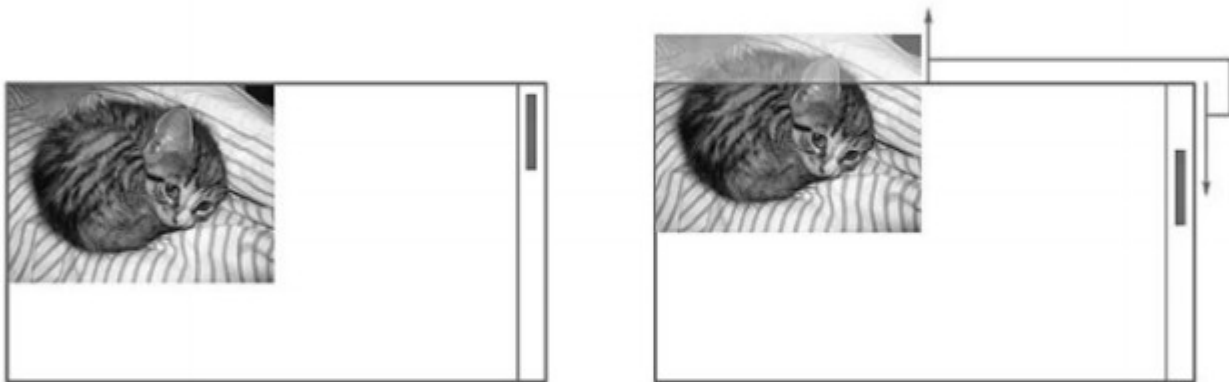
обычно заполняет всю область сплошным цветом. Но множественные фоны требуют прозрачности хотя бы в одном фоне. Таким образом, это свойство не может быть использовано в случае множественных фонов.

Но это еще не все, что можно сказать о фоновых изображениях. Закончим наше обсуждение, рассмотрев другие случаи.

14.6. Свойство `background-attachment`

Можно определить поведение фонового изображения относительно полосы прокрутки.

`background-attachment: scroll:`



Фиксированные фоны не реагируют на полосу прокрутки.

`background-attachment: fixed:`

