6 Форматирование текста

Большинство сайтов во Всемирной паутине по-прежнему построены исключительно на текстовом контенте. Несомненно, людям нравится видеть фотографии, видеоклипы, анимацию, но все-таки именно содержательный текстовый материал заставляет их вновь и вновь возвращаться на определенные сайты. Люди жаждут обновлений в социальных сетях типа Facebook, новостей, статей с практическими рекомендациями, рецептов, ответов на актуальные вопросы, полезной информации и новых записей в микроблогах Twitter. С помощью языка CSS вы можете и *должны* придать такой вид заголовкам и абзацам веб-страниц, что они привлекут внимание посетителей не хуже фотографий. Оформление текстового контента называется *типографикой*.

Каскадные таблицы стилей предлагают для этих целей обширный набор мощных команд форматирования, которые позволяют назначать шрифты, цвет, кегль, межстрочный интервал и другие свойства и атрибуты, оказывающие влияние на визуальное восприятие как отдельных элементов веб-страницы (заголовков, маркированных списков, обычных абзацев текста), так и всей веб-страницы, сайта в целом (рис. 6.1). Настоящая глава описывает и показывает все многообразие свойств форматирования текстового содержимого веб-страниц и заканчивается практикумом, позволяющим поупражняться в создании таблиц стилей для текста и применении их к реальным веб-страницам.

Использование шрифтов

Первое, что вы можете сделать для увеличения привлекательности сайта, — применить различные шрифты к заголовкам, абзацам и другим элементам. Для выбора начертания в каскадных таблицах стилей используется свойство `font-family` и указывается тот шрифт, который следует применять. Предположим, вы хотите определить для абзацев шрифт Arial. Для этого можно создать селектор тега для элемента `p` и воспользоваться свойством `font-family`:

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

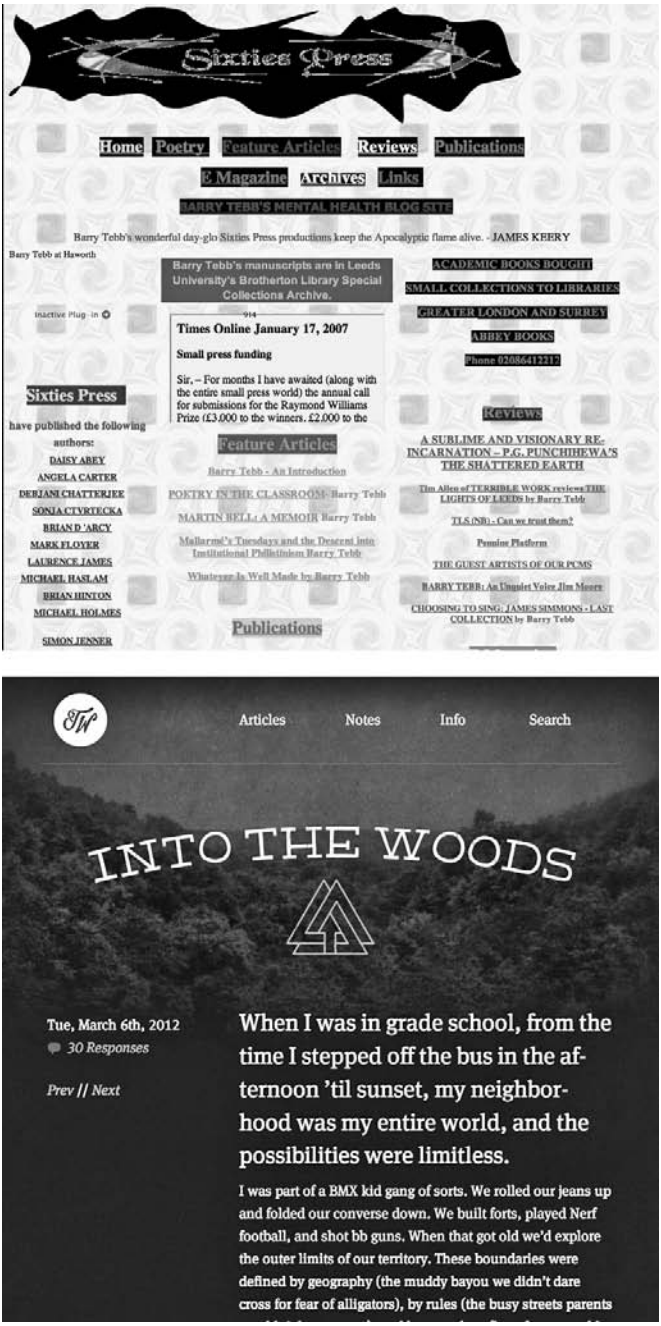

Рис. 6.1. Использование слишком большого количества шрифтов или типографических украшений зачастую приводит читателя в замешательство, затрудняя понимание сути информации, изложенной на веб-странице *(вверху)*. Использование текста со шрифтом различных размеров, умелый подбор стиля и всего пара разных шрифтов облегчает просмотр страницы и делает ее приятной для чтения *(внизу)*

Изначально свойство `font-family` функционирует, только если у посетителей сайта имеется такой же шрифт, установленный на их компьютерах. Иначе говоря, применительно к показанному выше примеру, если у кого-то из посетителей вашего сайта не будет на компьютере установлен шрифт Arial, абзацы страницы будут отображены с использованием исходного шрифта браузера (обычно это какой-нибудь вариант Times New Roman). Поэтому веб-дизайнеры ограничивались небольшим набором шрифтов, предустановленным на большинстве компьютеров.

В последнее время браузеры стали поддерживать *веб-шрифты*, то есть такие шрифты, которые браузер загружает и применяет при просмотре вашего сайта. Веб-шрифты также используют свойство `font-family`, но требуют набора дополнительного CSS-кода, называемого правилом `@font-face` — оно инструктирует браузер загрузить указанный шрифт. Веб-шрифты открывают многие захватывающие возможности в области дизайна, позволяя выбирать из бурно растущего количества гарнитур.

Но вскоре мы увидим, что они также приносят свой набор проблем. Иначе говоря, как веб-дизайнер вы можете остановиться на выборе проверенных и реально существующих шрифтов, то есть выбрать шрифты из числа установленных на большинстве компьютеров, или же выбрать веб-шрифты, расширив свой дизайнерский кругозор (ценой дополнительной работы). Но вы не ограничены только одним из этих подходов. Многие дизайнеры используют их в сочетании, в одних случаях применяя стандартные шрифты (например, для абзацев основного текста страницы), а в других — веб-шрифты (например, для создания привлекательных заголовков).

Выбор подходящего шрифта

Если для указания шрифта используется свойство `font-family`, посетители могут не увидеть выбранный вами шрифт, так как у них он либо должен быть уже установлен на компьютере, либо, при указании веб-шрифтов, временно загружен в кэш браузера. Поскольку вы не можете знать, доступен ли предпочитаемый вами шрифт конкретному пользователю, сложилась практика указывать не только основной шрифт, но и пару резервных вариантов. Этот список называется *стеком шрифтов*. Если на компьютере посетителя сайта установлен (доступен) шрифт, выбранный первым, он и будет использоваться для форматирования текста. Но если первый шрифт не установлен, то браузер просматривает список, пока не обнаружит на компьютере указанный в стеке шрифт. Идея состоит в том, чтобы определить список похожих шрифтов, присутствующих в большинстве операционных систем. Например:

```
font-family: Arial, Helvetica, sans-serif;
```

В данном примере браузер сначала выяснит, установлен ли на компьютере шрифт Arial. Если да, то браузер использует этот шрифт. В противном случае он ищет шрифт Helvetica и, если и тот не установлен, применяется последний из списка — универсальный рубленый (также называется гротеском, шрифтом без засечек) шрифт семейства sans-serif. Если вы вносите в список универсальный тип шрифта (sans-serif или serif), то браузер выбирает установленный на компьютере шрифт из этого семейства. По крайней мере, таким образом вы можете определить его основной символ.

ПРИМЕЧАНИЕ

На практике для применения определенного стиля вы должны добавить селектор и фигурные скобки. Например:

p { font-family: Arial, Helvetica, sans-serif; }

Если в этой книге вы встретите примеры вида font-family: Arial, Helvetica, sans-serif;, помните, что это всего лишь отдельное свойство CSS-стиля для сокращения и удобства чтения.

Если имя шрифта состоит из нескольких слов, вам следует заключать его в кавычки:

```
font-family: "Times New Roman", Times, serif;
```

Далее представлены некоторые часто используемые сочетания обычно установленных шрифтов, сгруппированные по типу шрифта (каждый список заканчивается универсальным типом шрифта).

Антиквенные шрифты

Антиквенные шрифты (с засечками) идеальны для длинных фрагментов текста, поскольку распространено мнение, что засечки — маленькие росчерки на концах основных штрихов — хорошо для глаз связывают одну букву с другой, делая текст более читабельным. Примерами антиквенных шрифтов являются Times, Times New Roman, Georgia.

- `"Times New Roman", Times, serif.`
- `Georgia, "Times New Roman", Times, serif.`
- `Baskerville, "Palatino Linotype", Times, serif.`
- `"Hoefler Text", Garamond, Times, serif.`

Примеры этих шрифтов приведены на рис. 6.2.

Сглаживание экранных шрифтов в операционной системе OS X выполняется иначе, чем в Windows. Операционная система Windows использует технологию ClearType, которая позволяет улучшить отображение текста на экране. Вид экранного текста в Windows зависит от используемых настроек ClearType. Подробнее о технологии ClearType можно узнать по адресу tinyurl.com/pmzxbnm.

СОВЕТ

На сайте cssfontstack.com опубликован список шрифтов, по умолчанию установленных в операционных системах OS X и Windows, включая подробное процентное соотношение использования шрифтов для каждой из них. Например, шрифт Courier New установлен на 99,73 % компьютеров, работающих под управлением операционной системы Windows и 95,68 % компьютеров Mac.

Рубленые шрифты

Рубленые шрифты (без засечек) часто используются для заголовков благодаря простому и четкому внешнему виду. Примеры рубленых шрифтов — Arial, Helvetica и Verdana.

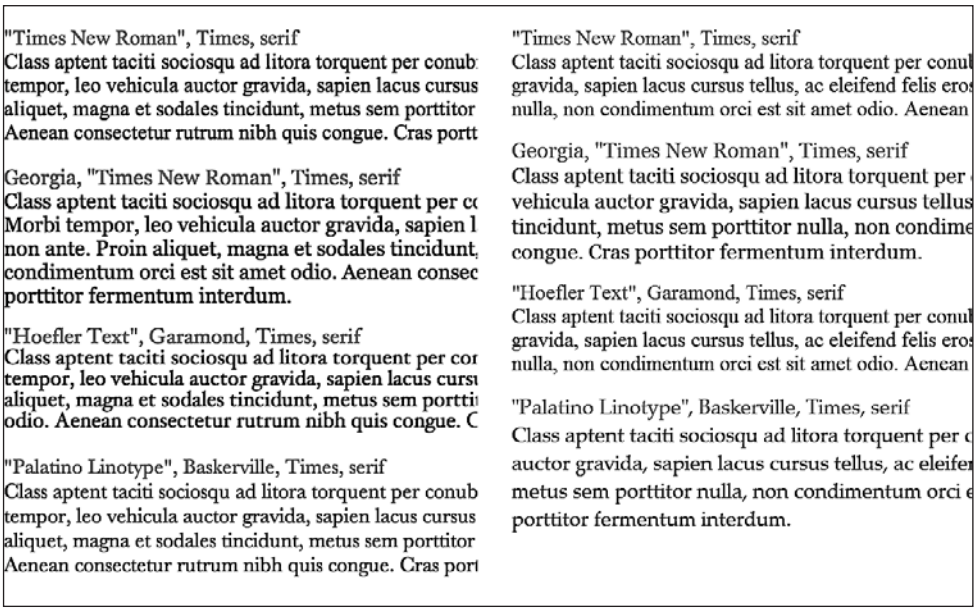

Рис. 6.2. Шрифты не всегда отображаются одинаково в операционных системах Windows *(справа)* и OS X *(слева)*. У этих двух систем различные наборы предустановленных шрифтов

- `Arial, Helvetica, sans-serif`.
- `Verdana, Arial, Helvetica, sans-serif`.
- `Geneva, Arial, Helvetica, sans-serif`.
- `Tahoma, "Lucida Grande", Arial, sans-serif`.
- `"Trebuchet MS", Arial, Helvetica, sans-serif`.
- `"Century Gothic", "Gill Sans", Arial, sans-serif`.

Примеры данных шрифтов приведены на рис. 6.3.

Некоторые люди верят, что на веб-страницах нужно использовать лишь рубленые шрифты, потому что декоративные штрихи антиквенных шрифтов не очень хорошо отображаются на экранах с низким разрешением. Тем не менее современные мониторы с высоким разрешением, которые отображают большее количество пикселов на дюйм, избавлены от этой проблемы. В конце концов, ваш вкус — лучший ориентир. Выбирайте те шрифты, которые считаете нужными.

Моноширинные и декоративные шрифты

Моноширинный шрифт часто используется для отображения компьютерного кода (например, фрагментов кода повсюду в этой книге). Каждая буква в моноширинном шрифте имеет одинаковую ширину (как буквы в механических печатных машинках).

- `"Courier New", Courier, monospace`.
- `"Lucida Console", Monaco, monospace`.

- `"Copperplate Light", "Copperplate Gothic Light", serif`.
- `"Marker Felt", "Comic Sans MS", fantasy`.

С примерами этих списков шрифтов можете ознакомиться на рис. 6.4.

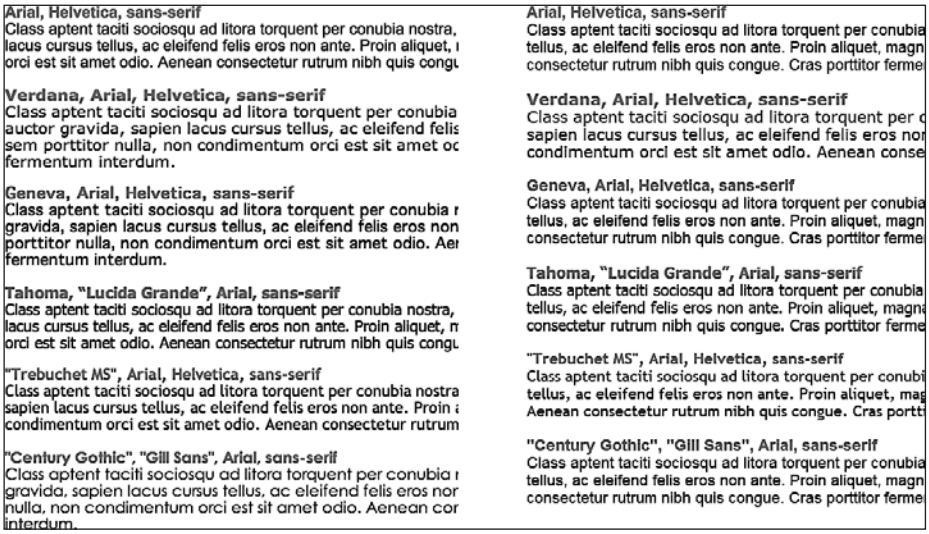

Рис. 6.3. Текст, оформленный рублеными шрифтами в операционных системах Windows *(справа)* и OS X *(слева)*

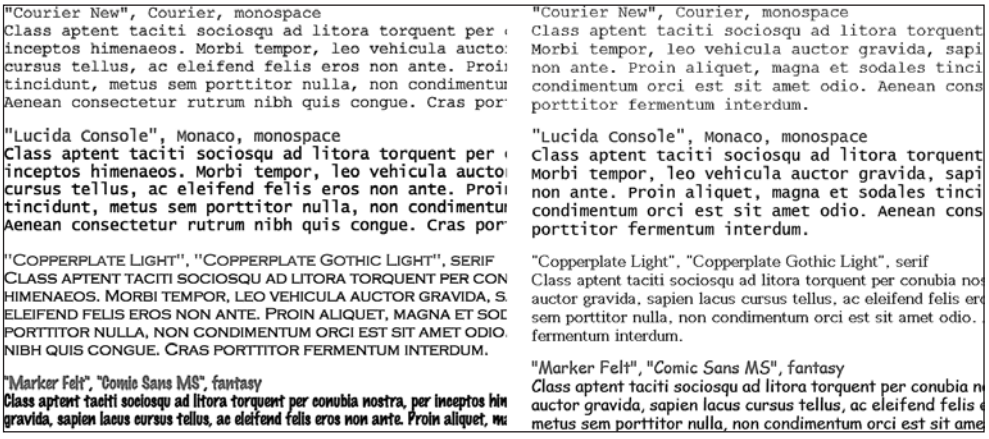

Рис. 6.4. Пример использования моноширинного шрифта в операционных системах Windows *(справа)* и OS X *(слева)*. Courier New — самый распространенный моноширинный шрифт, но ограничиваться только им не обязательно. Шрифт Lucida Console очень популярен в Windows и OS X, а Monaco установлен на каждом компьютере Mac

Дополнительные шрифты

На самом деле существуют буквально тысячи шрифтов, и каждая операционная система поставляется со многими из тех, которые не перечислены здесь. Тем не

менее приведу несколько шрифтов, которые очень часто встречаются на персональных компьютерах и компьютерах OS X. Возможно, вы захотите выбрать какие-то из них:

- Arial Black;
- Arial Narrow;
- Impact.

Будьте осторожны с Arial Black и Impact: у них есть только одно начертание и они не поддерживают курсивный вариант. Соответственно, если вы используете эти шрифты, не забудьте присвоить свойствам `font-weight` и `font-style` значение `normal`. В противном случае, если шрифт будет полужирным или курсивным, браузер сделает все от него зависящее, чтобы текст выглядел ужасно.

Использование веб-шрифтов

Традиционный способ применения шрифтов в каскадных таблицах стилей прост и понятен: нужно указать желаемый шрифт, воспользовавшись свойством `font-family`. Но вы ограничены теми шрифтами, которые установлены на компьютерах посетителей вашего сайта. На этот случай, как уже ранее упоминалось, все основные браузеры поддерживают использование веб-шрифтов. При этом браузеры фактически загружают шрифт с веб-сервера и используют его для отображения текста на веб-странице.

Процедура использования веб-шрифтов довольно проста. Вам потребуется лишь:

- правило `@font-face`, отвечающее за сообщение браузеру как имени шрифта, так и пути, по которому его нужно загрузить. Вскоре вы узнаете, как это правило используется, а сейчас имейте в виду, что с его помощью браузеру сообщается о необходимости загрузить шрифт;
- свойство `font-family`, которое применяется с веб-шрифтами точно так же, как и со шрифтами, установленными на компьютере. Другими словами, если есть правило `@font-face`, инструктирующее браузер загрузить шрифт, вы можете назначить этот шрифт любому стилю CSS с помощью свойства `font-family`.

Теоретически веб-шрифты использовать не трудно. Но если вдаваться в подробности, то для их правильного применения нужно понимать суть некоторых специфических требований.

ПРИМЕЧАНИЕ

Довольно простой способ использования веб-шрифтов предлагается компанией Google. Вскоре мы рассмотрим, в чем именно он заключается.

Типы файлов шрифтов

Не верится, но браузер Internet Explorer поддерживал веб-шрифты, начиная с 5-й версии программы (выпущенной более 15 лет назад!). Но эта поддержка требовала использования весьма специфического и сложного способа создания типографиче-

ского форматирования. Иными словами, нельзя было взять обычный шрифт с жесткого диска компьютера, выложить его на веб-сервер и считать, что дело сделано. Сначала шрифт нужно было преобразовать в формат EOT (Embedded Open Type — встраиваемый формат Open Type). И такое положение вещей сохранялось вплоть до версии Internet Explorer 8.

Для веб-шрифтов используются и другие форматы шрифтов, причем разные браузеры поддерживают различные форматы. Чтобы указанными шрифтами могло любоваться как можно большее количество посетителей вашего сайта, нужно предоставить эти шрифты в различных форматах (как это сделать, вы вскоре узнаете).

В следующем списке приведены различные типы шрифтов и перечислены браузеры, которые с ними работают.

- **EOT.** Шрифты Embedded Open Type поддерживаются только в браузере Internet Explorer. Чтобы преобразовать обычный шрифт в формат EOT, потребуется специальное программное обеспечение, но сделать это можно и на таких сайтах, как FontSquirrel.
- **True Type и Open Type.** Если заглянуть в папку Fonts вашего компьютера, то там наверняка найдутся файлы шрифтов с расширениями `.ttf` (True Type) или `.otf` (Open Type). Шрифты этих форматов чаще всего используются на компьютерах. Их можно задействовать для обработки текста и вывода его на экран, а также для веб-страниц. Ранее это были одни из самых используемых веб-шрифтов, и они все еще поддерживаются большинством браузеров. Однако им на смену пришел формат WOFF.
- **WOFF.** Формат Web Open Font был разработан специально для Всемирной паутины. WOFF-шрифты, по сути, являются сжатой версией шрифтов TrueType или Open Type. Это означает, что WOFF-шрифты обычно имеют меньший размер файла и загружаются быстрее других форматов. Формат WOFF имеет также широкую поддержку со стороны браузеров, включая Internet Explorer 9 и выше, Firefox, Chrome, Safari, Opera, а также браузеры в операционной системе iOS, Blackberry и Android версии 4.4 и выше.

ПРИМЕЧАНИЕ

WOFF2 — последняя версия формата WOFF, обеспечивающего на 30 % более эффективное сжатие, благодаря чему файлы шрифтов становятся меньшего размера и загружаются быстрее. Однако на момент написания этой книги данный формат не поддерживался браузерами Internet Explorer, Edge, Safari и Opera Mini (tinyurl.com/pucouja).

- **SVG.** Scalable Vector Graphic (формат масштабируемой векторной графики) сам по себе не является форматом шрифта. По сути, это способ создания *векторной графики* (то есть изображений, которые могут масштабироваться без потери качества). Поддержка SVG-шрифтов ограничена куда существеннее. SVG-формат поддерживается только браузером Safari. Другой проблемой, связанной с SVG, является то, что данный формат создает файлы размером в два раза больше, чем TrueType-файлы, и в три раза больше, чем WOFF-файлы. Единственное реальное преимущество SVG в том, что только этот формат шрифта поддерживают устаревшие версии операционной системы iOS с браузером Safari версии 4.1 или ниже, а также браузеры в операционной системе

Android 4.3 и ниже. Если вы не предполагаете просмотр ваших веб-страниц на этих устройствах, не используйте формат SVG.

Вам не нужно выбирать только один формат шрифтов, игнорируя все остальные, не поддерживающие его браузеры. Совсем скоро вы узнаете, что можно (и, как правило, нужно) указывать несколько форматов, позволяя браузеру выбирать только тот, который он поддерживает. Кроме того, вы можете загрузить шрифт, который уже был преобразован в эти четыре формата, или даже преобразовать обычный шрифт True Type в несколько форматов.

ПРИМЕЧАНИЕ

Каждый файл шрифта содержит только одно начертание. Иначе говоря, если вам нужны и полужирный и курсивный начертания, следует загрузить отдельные файлы шрифта для каждого начертания. Некоторые шрифты, обычно декоративные, могут существовать только в одном начертании и больше подходят для заголовков или прочего текста, где не нужно курсивное или полужирное форматирование. Более полная информация рассмотрена далее в этой главе.

Правовые вопросы использования веб-шрифтов

Вторым препятствием для использования веб-шрифтов являются правовые вопросы. Эти шрифты с целью получения средств к существованию создаются и продаются как отдельными разработчиками, так и компаниями. После загрузки шрифта TrueType на ваш сервер для его использования посетителями при просмотре вашего сайта любой человек может скачать шрифт и пользоваться им на своем сайте или в установленных на его компьютере программах допечатной подготовки или текстовых редакторах. Большинству компаний, занимающихся созданием шрифтов, не нравится нелегальное использование плодов их труда, поэтому многие шрифты имеют лицензии, которые конкретно запрещают их использование во Всемирной сети.

Иными словами, даже если вы приобрели шрифт в корпорации Adobe, его нельзя использовать на вашем сайте. Многие компании, создающие шрифты, в настоящее время предлагают различные виды лицензий (по разным ценам), чтобы разрешить использование своих шрифтов во Всемирной паутине. Это касается даже шрифтов, поставляемых вместе с вашим компьютером. Вам разрешается использовать их с программами, установленными на компьютере, но может быть не разрешено помещать файлы тех же самых шрифтов на свой веб-сервер для использования их в качестве веб-шрифтов. Если вы не знаете, разрешено ли применение шрифта в Сети, лучше воздержаться от его использования и подобрать шрифт, который в ней может применяться легально.

ПРИМЕЧАНИЕ

Чтобы решить возникающие правовые вопросы, можно воспользоваться такими службами шрифтов, как Google Fonts или TypeKit, коммерческой службой веб-шрифтов корпорации Adobe (о которой будет рассказано в одной из следующих врезок).

Поиск веб-шрифтов

При поиске веб-шрифтов приходится сталкиваться с двумя проблемами: поиском таких шрифтов, которые легально можно использовать во Всемирной паутине, и по-

иском форматов шрифтов, которые понадобятся посетителям вашего сайта (EOT, WOFF, TrueType и SVG). Хотя некоторые компании стали предлагать лицензии, разрешающие использование приобретаемых шрифтов во Всемирной паутине, существует довольно широкий ассортимент бесплатных веб-шрифтов. Ниже представлены лишь некоторые из множества источников бесплатных веб-шрифтов.

- tinyurl.com/olkcst6. Представленный группой дизайнеров, этот сайт был одним из первых, предложивших бесплатное использование созданных вручную шрифтов во Всемирной паутине. Созданный ими шрифт League Gothic широко применяется на сайтах.
- exljbris.com. Предоставляет классические бесплатные шрифты: Museo, Museo Sans и Museo Slab.
- openfontlibrary.org. Доступно более 700 бесплатных шрифтов (на момент написания книги), и все они могут использоваться на ваших сайтах.
- fontsquirrel.com. Весьма примечательный сайт в мире веб-шрифтов, предлагающий более тысячи шрифтов. В дополнение к шрифтам ресурс предлагает инструментарий для преобразования шрифта TrueType или Open Type в другие форматы, включая EOT, SVG и WOFF. Порядок использования этого инструментария будет рассмотрен в следующем разделе.
- google.com/webfonts. Компания Google предоставляет простой и бесплатный способ включения веб-шрифтов в ваши сайты. Порядок использования этой службы будет подробно рассмотрен чуть позже.

Преобразование форматов шрифтов

Большинство сайтов, предлагающих бесплатные шрифты, предоставляют шрифт в единственном формате, обычно TrueType (`.ttf`) или Open Type (`.otf`). Но TrueType и Open Type некоторыми браузерами не поддерживаются. Кроме того, формат WOFF поддерживается всеми современными браузерами и имеет преимущество, так как файлы в этом формате меньше по размеру, чем TrueType. В настоящий момент новый формат WOFF2 поддерживается лишь несколькими браузерами, но в скором времени ситуация должна измениться.

Существует несколько моментов, о которых надо помнить при использовании веб-шрифтов на вашем сайте. Во-первых, вы можете быть консервативными и использовать шрифты, которые поддерживаются старыми браузерами и мобильными устройствами. То есть добавить шрифт в формате EOT (для браузера Internet Explorer 8 и версии ниже), шрифты TrueType для устаревших браузеров, шрифты WOFF для современных браузеров и шрифты SVG для старых смартфонов и планшетов.

Кроме того, поскольку формат WOFF поддерживается всеми современными браузерами, вы можете использовать *только* его. В этом случае старые браузеры при просмотре страниц будут пропускать шрифт WOFF и использовать вместо него следующий в стеке шрифтов (см. предыдущий раздел).

И наконец, вы можете использовать только шрифты EOT и WOFF. Файл EOT предназначен для браузера Internet Explorer 8, который все еще применяется. Файл WOFF будут использовать остальные современные браузеры (включая браузер IE 9 и более поздние версии программы).

Для продолжения занятия вам необходимо создать файлы шрифтов. Для этих целей ресурс Font Squirrel предоставляет очень полезное средство, помогающее преобразовывать файлы шрифтов в нужные форматы. Инструмент Webfont Generator, доступный по адресу tinyurl.com/b56z7zq, является простым средством создания не только нужных шрифтов, но и пробных HTML-файла и таблицы стилей.

Чтобы воспользоваться им, выполните следующие действия.

1. Найдите шрифт TrueType (`.ttf`) или Open Type (`.otf`).

   Воспользуйтесь одним из сайтов, перечисленных в предыдущем разделе, или найдите шрифт на своем компьютере. Убедитесь, что этот шрифт лицензирован для использования в качестве веб-шрифта. Если он не лицензирован или вы не уверены в его лицензировании, найдите другой шрифт.

2. Перейдите к инструменту Webfont Generator по адресу tinyurl.com/b56z7zq.

   Это простая страница, на которой находится лишь несколько настроек (рис. 6.5).

3. Нажмите кнопку Upload fonts (Выгрузить шрифты) (см. рис. 6.5, *1*).

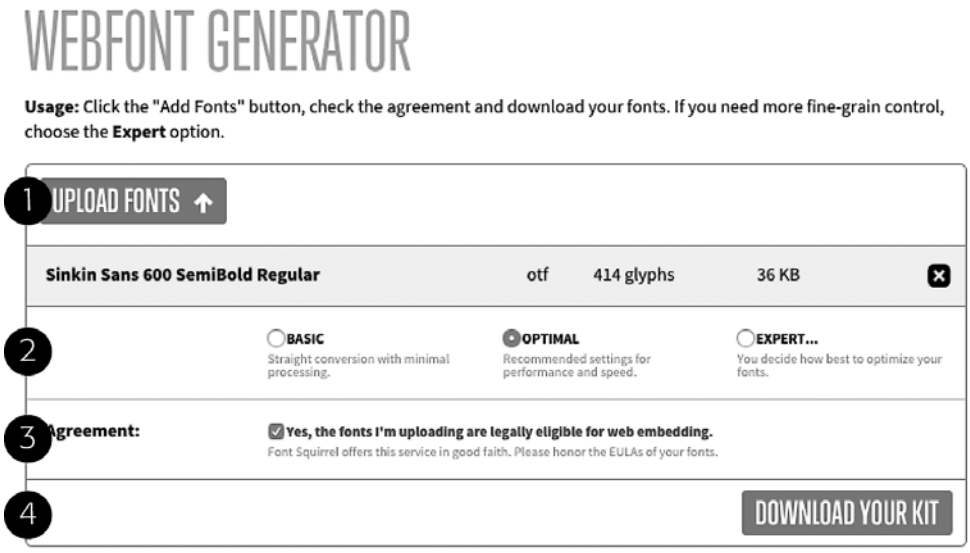

Рис. 6.5. Окно генератора шрифтов

Браузер откроет диалоговое окно Выгрузка файла (Select files).

4. Выберите один или несколько шрифтов на жестком диске компьютера и нажмите кнопку Открыть (Open).

   Файл (файлы) будет загружен на сервер fontsquirrel.com.

5. Выберите вариант преобразования (см. рис. 6.5, *2*).

   - Basic (Простое) — будет выполнено только преобразование шрифта в форматы EOT, WOFF и SVG.

- Optimal (Оптимальное) — более удачный выбор, поскольку это не только приведет к преобразованию шрифта, но и будут выполнены другие действия по оптимизации форматов для повышения производительности и скорости загрузки.
- Expert (Экспертное) позволит провести тонкую настойку всех нюансов преобразования. Например, вы сможете настроить *набор символов*. Иными словами, из шрифта можно исключить определенные, неиспользуемые вами символы, например точку с запятой, восклицательный знак, букву Q или буквы с диакритическими знаками (такие как «ь», «й» или «х»). Кроме того, вы можете использовать экспертный режим, чтобы получить только определенные форматы шрифтов, такие как WOFF или WOFF2.

Пока вы не станете понимать смысл своих действий, придерживайтесь варианта настройки Optimal (Оптимальное). При выборе этого режима будут сгенерированы все необходимые вам форматы шрифтов и выполнены другие настройки, чтобы шрифты быстрее загружались и лучше выглядели на экране. А для полного контроля процесса выберите вариант Expert (Экспертное), чтобы отобразить перечень элементов управления, которыми можно воспользоваться для изменения настроек процесса создания шрифтов.

По всей вероятности, вам потребуется доступ ко всем символам, поэтому лучше остановить выбор на варианте Optimal (Оптимальное).

6. Установите флажок, обозначенный пунктом *3* на рис. 6.5.

   Как уже упоминалось, шрифты — это интеллектуальная собственность и их простое размещение на веб-сервере недопустимо. Убедитесь в том, что ваши шрифты могут использоваться во Всемирной паутине, прежде чем устанавливать флажок.

7. Нажмите кнопку Download Your Kit (Загрузить свой комплект) (см. рис. 6.5, *4*).

   В зависимости от количества преобразуемых шрифтов и их сложности, процесс может занять некоторое время. Серверу fontsquirrel.com нужно получить шрифты и преобразовать их в каждый выбранный формат. Как только все будет готово, вы сможете скачать архив, содержащий файлы различных форматов шрифтов, а также демонстрационные HTML- и CSS-файлы. Наиболее важными, конечно же, являются файлы шрифтов с расширениями `.eot`, `.ttf`, `.woff` и `.svg`, которые вы хотите использовать.

После получения шрифтов настало время научиться использовать их с правилом `@font-face`.

Использование правила @font-face

Сначала скопируйте шрифты в папку на компьютере, где хранятся файлы для вашего сайта. Многие веб-разработчики создают для этого специально выделенную папку в корневом каталоге сайта, которая носит имя fonts, _fonts или webfonts. В качестве альтернативы, если есть папка для CSS-файлов, вы можете поместить файлы шрифтов в эту папку. В принципе, неважно, куда вы поместите эти файлы на вашем сайте, но вышеописанное правило поможет навести в нем порядок.

Секретом использования веб-шрифтов является *правило* `@font-face`. Эта команда по своей сути присваивает шрифту имя и сообщает браузеру, где найти файл шрифта для его загрузки. Правило `@font-face` помещается в вашу таблицу стилей так же, как и обычный стиль. Предположим, вы используете шрифт League Gothic. У вас на сайте в папке fonts есть шрифт True Type с именем файла `League_Gothic-webfont.ttf`. Нужно инструктировать браузер загрузить этот шрифт, добавив правило `@font-face` в таблицу стилей:

```
@font-face {
  font-family: "League Gothic";
  src: url('fonts/League_Gothic-web font.woff');
}
```

Первое свойство, `font-family`, вы уже встречали, но здесь у него другое предназначение. При использовании внутри правила `@font-face` свойство `font-family` присваивает шрифту имя. Затем это имя задействуется, когда соответствующий шрифт нужно применить к стилю. Предположим, нужно использовать шрифт League Gothic для всех абзацев на странице. Для этого можно будет указать следующий стиль:

```
p {
  font-family: "League Gothic";
}
```

ПРИМЕЧАНИЕ

Для каждого шрифта, который нужно использовать, применяется отдельное правило @font-face. Если у вас задействованы три шрифта, понадобятся три правила @font-face. Лучше всего использовать их сгруппированными вместе и помещать в верхнюю часть таблицы стилей, чтобы браузер сразу же мог начать загрузку необходимых файлов шрифтов.

Второй атрибут, свойство `src`, сообщает браузеру, где следует искать файл шрифта на сервере. Путь к файлу шрифта помещается в кавычки внутрь атрибута `url()`. Путь к шрифту указывается точно так же, как и все другие пути, например к изображениям, ссылкам и JavaScript-сценариям. Предположим, у вас есть таблица стилей в папке с именем _styles и есть файл шрифта `my_font.ttf` в папке _fonts. Обе папки находятся в корневом каталоге вашего сайта. Следовательно, путь из таблицы стилей к файлу шрифта будет выглядеть так: `../_fonts/my_font.ttf`. И тогда для этого шрифта нужно будет написать следующее правило `@font-face`:

```
@font-face {
  font-family: "Название шрифта";
  src: url('../_fonts/my_font.woff');
}
```

Возможно, вы заметили, что в приведенном выше примере фигурирует только один файл шрифта в формате WOFF. Это сделано для упрощения демонстрации общего принципа работы правила `@font-face`. Как уже говорилось, оно позволяет указать несколько файлов шрифтов различных форматов.

Если вы хотите обеспечить поддержку старых браузеров, смартфонов и планшетов, то синтаксис должен быть немного усложнен. Предположим, на сайте нужно использовать все разновидности форматов шрифта League Gothic. Тогда показанный выше код нужно переписать следующим образом:

```
@font-face {
  font-family: 'League Gothic';
  src: url('fonts/League_Gothic-webfont.eot');
  src: url('fonts/League_Gothic-webfont.eot?#iefix') format('embeddedopentype'),
       url('fonts/League_Gothic-webfont.woff2') format('woff2'),
       url('fonts/League_Gothic-webfont.woff') format('woff'),
       url('fonts/League_Gothic-webfont.ttf') format('truetype'),
       url('fonts/League_Gothic-webfont.svg') format('svg');
}
```

Код неоправданно усложнен из-за несовместимости браузера Internet Explorer. Давайте в нем разберемся.

- Строка 2 сохранила свой прежний вид. Свойство `font-family` предоставляет имя шрифта, это то же самое имя, которое вы будете использовать, применяя шрифт к своим стилям.
- Строка 3 предназначена для программы Internet Explorer 9, но только при ее работе в режиме совместимости — когда IE 9 работает как IE 8. Это странное свойство было добавлено в IE 9, чтобы сайты, адаптированные под недостатки IE 8 и более ранних версий, не теряли в дизайне и в IE 9. Пользователь должен будет преднамеренно переключиться в режим совместимости в IE 9, поэтому лучше, наверное, оставить эту настройку.
- Строка 4 начинается со второго свойства `src`, которое в соответствии с правилом `@font-face` может указывать на несколько шрифтов. Первым вновь указан шрифт `.eot`, но на этот раз к расширению файла добавлена строка `?#iefix`. Это сделано в попытке приспособиться к дополнительным недостаткам Internet Explorer, на этот раз дело касается версий 6–8 браузера. Если после расширения `.eot` не будет указано это значение, то шрифт может неправильно отобразиться в Internet Explorer 8 и более ранних версиях программы.

  После URL-адреса можно также заметить новый фрагмент кода:

  ```
  format('embedded-opentype')
  ```

  Он указывает формат шрифта и добавляется после каждого URL-адреса различных форматов шрифтов.
- Строки 5–8 определяют дополнительные форматы шрифтов. Приведено всего одно свойство (`src`) на нескольких строках для упрощения чтения кода. Для каждого формата шрифта, указанного в свойстве `src`, добавляется URL-адрес, атрибут `format` и запятая (для всех, за исключением последнего шрифта):

  ```
  url('fonts/League_Gothic-web font.woff') format('woff'),
  ```

ПРИМЕЧАНИЕ

В конце списка файлов свойства src добавляется точка с запятой, чтобы обозначить его конец (строка 7 в показанном выше примере). Об этой завершающей точке с запятой забывать нельзя, иначе правило @font-face работать не будет.

Даже если браузер поддерживает различные форматы шрифтов (например, программа Chrome может использовать шрифты WOFF2, WOFF, TrueType и SVG), он не станет загружать все файлы шрифтов. Вместо этого по мере считывания

списка форматов шрифтов браузер загружает только первый поддерживаемый шрифт. Иначе говоря, если браузеру Chrome попадется показанный выше код, он пропустит файл с расширением `.eot`, поскольку этот формат им не поддерживается, а загрузит файл с расширением `.woff`. Затем он полностью проигнорирует файлы форматов TrueType и SVG. Из этого следует, что порядок, в котором перечислены шрифты, играет важную роль. WOFF в большинстве случаев предпочтительнее, поскольку файлы этого формата меньше по размеру и загружаются быстрее. WOFF2 — это оптимизированная версия формата WOFF, которая обеспечивает использование файлов меньшего размера, но в настоящий момент поддерживается не всеми браузерами. Файлы формата SVG намного больше по размеру и поддерживаются только браузером Safari. Подводя итог вышесказанному, необходимо отметить следующее: для того чтобы браузеры сначала загружали файлы шрифтов меньших размеров, нужно убедиться, что шрифты перечислены в следующем порядке: `.eot`, `.woff2`, `.woff`, `.ttf` и `.svg`.

Если вы хотите обеспечить поддержку браузера Internet Explorer 8 и более современных браузеров, то можете использовать только форматы EOT, WOFF2 и WOFF. Фактически, если вас не интересует браузер IE 8 (см. врезку «Стоит ли волноваться насчет Internet Explorer 8?» в главе 1), вы можете использовать только шрифты формата WOFF. Следующие два листинга демонстрируют альтернативные способы применения правила `@font-face`:

```
@font-face {
  font-family: 'League Gothic';
  src: url('fonts/League_Gothic-webfont.eot?#iefix') format('embeddedopentype'),
       url('fonts/League_Gothic-webfont.woff2') format('woff2'),
       url('fonts/League_Gothic-webfont.woff') format('woff');
}
```

Если вас не интересует браузер Internet Explorer 8 и его более ранние версии, вы можете упростить событие следующим образом:

```
@font-face {
  font-family: 'League Gothic';
  src: url('fonts/League_Gothic-webfont.woff2') format('woff2'),
       url('fonts/League_Gothic-webfont.woff') format('woff');
}
```

В этой книге используется последний вариант, предназначенный только для современных браузеров. Но, если вы предполагаете, что посетители вашего сайта будут работать с Internet Explorer 8, используйте версию правила, включающую файлы `.eot`. Аналогичным образом, если вам необходима поддержка старых смартфонов и планшетов, перечисленных выше, вам лучше использовать более сложный синтаксис, показанный на с. 155.

ПРИМЕЧАНИЕ

Не стоит волноваться, что, если ваш сайт не содержит веб-шрифты, предназначенные для старых браузеров (таких форматов, как .svg, .ttf, или .eot), то посетители не смогут просмотреть его контент. Помните, что при определении шрифтов вы используете стек шрифтов, поэтому браузеры, не поддерживающие указанный в правиле шрифт, будут применять похожую гарнитуру. В большинстве случаев ваш сайт будет выглядеть одинаково, независимо от браузера.

Создание стилей с применением веб-шрифтов

Самым сложным в работе с веб-шрифтами является получение файлов шрифтов в нужном формате и настройка правила `@font-face`. После того как все это будет готово, веб-шрифты используются точно так же, как и рассмотренные ранее шрифты, предустановленные на компьютере (устройстве). Иначе говоря, при создании стиля применяется свойство `font-family` и указывается имя шрифта, которое добавлено в правиле `@font-face`. Например, в предыдущем коде в правиле `@font-face` шрифту было присвоено имя League Gothic (строка 2). Это имя и нужно использовать, применяя данный шрифт к стилю. Чтобы для всех заголовков `h1` устанавливался шрифт League Gothic, можно создать следующий стиль:

```
h1 {
  font-family: 'League Gothic';
  font-weight: normal;
}
```

Обратите внимание на новое свойство `font-weight`. Обычно браузеры отображают содержимое элементов `h1` полужирным шрифтом. Большинство из них искусственно сделают шрифт полужирным, когда требуется такая версия шрифта. В результате получается некрасивое полужирное начертание. Присваивание свойству `font-weight` значения `normal` инструктирует браузер использовать шрифт League Gothic таким, какой он есть, и пресекает попытки сделать его полужирным. Более подробно работа с начертаниями применительно к веб-шрифтам будет рассмотрена в следующем разделе.

Рекомендуется также включить список резервных предустановленных шрифтов на тот случай, если браузер не сможет загрузить веб-шрифт. Здесь можно воспользоваться ранее рассмотренной технологией. Например:

```
h1 {
  font-family: 'League Gothic', Arial, sans-serif;
  font-weight: normal;
}
```

СОВЕТ

На веб-странице можно также задействовать шрифты, содержащие различные символы и значки. Иначе говоря, вместо создания, к примеру, рисунка предупреждения и помещения его в текст абзаца можно воспользоваться правилом @font-face для загрузки шрифта, содержащего значок аналогичного текстового знака, и применить соответствующий символ. Тем не менее предварительно проверьте стандартные символы Unicode или задумайтесь об использовании SVG-графики для представления значка. Подробнее тема освещается на сайте tinyurl.com/oaclzaq.

Форматирование полужирным и курсивным начертаниями

Обычные шрифты, установленные на компьютере, включают варианты начертания и веса (насыщенности), поэтому при применении в HTML-коде элемента `strong` браузер использует полужирное начертание такого шрифта, а при использовании элемента `em` — курсивное. Если же указывать эти два элемента совместно,

вы увидите полужирную курсивную версию шрифта. Это совершенно другие шрифты, содержащиеся в других файлах шрифтов. При исходном способе применения шрифтов на веб-страницах (который был рассмотрен ранее) вам не нужно волноваться насчет этих других шрифтов, потому что браузер автоматически воспользуется правильной версией.

А вот в случае с веб-шрифтами для каждого варианта шрифта вам понадобится отдельный файл. Итак, для основного текста веб-страницы нужны как минимум обычная версия шрифта, полужирная, курсивная, а также комбинированная полужирная и курсивная версия. И об этом нужно помнить при подборе шрифта для своего сайта, поскольку у некоторых шрифтов доступна только версия с определенным начертанием и нет, к примеру, курсивной версии. Такой шрифт пригодится для заголовков, но применять его для форматирования длинных текстовых абзацев, где, скорее всего, будут встречаться слова в полужирном и курсивном начертании, нет смысла. Кроме того, для каждого варианта шрифта вы должны создать отдельное правило `@font-face`.

При работе с курсивными и полужирными начертаниями веб-шрифтов доступно два варианта действий. Один из них проще реализовать, но он не работает в программе Internet Explorer 8 или более ранней версии (или же IE 9 в режиме совместимости), другой требует больших трудозатрат, но работает и в старых версиях браузера IE.

Простой способ добавления полужирного и курсивного начертания

Самый простой способ добавления полужирного и курсивного начертаний ваших шрифтов заключается в добавлении в правило `@font-face` свойств `font-weight` и `font-style`. Обычно свойство `font-weight` требует от браузера отобразить шрифт в полужирном (`bold`), обычном (`normal`) варианте или в одном из нескольких других вариантов веса, а свойство `font-style` управляет отображением шрифта в курсивном (`italic`) или обычном (`normal`) начертании. Но при использовании в правиле `@font-face` свойство `font-style` требует от браузера применить шрифт, когда стиль запрашивает конкретный вариант шрифта.

Предположим, у вас есть шрифт PTSans. Вы начинаете с обычной — не жирной и не курсивной — версии шрифта. Различные начертания шрифта начинаются с PTSansRegular. В таблицу стилей нужно добавить следующее правило `@font-face`:

```
@font-face {
  font-family: 'PTSans';
  src: url('PTSansRegular.woff2') format('woff2'),
       url('PTSansRegular.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}
```

ПРИМЕЧАНИЕ

В приведенном выше примере не используется шрифт в формате EOT, необходимый браузеру Internet Explorer 8, поскольку этот способ в нем не работает. Код из примера также игнорирует старые версии браузеров для компьютеров и мобильных устройств. Однако он прекрасно функционирует в современных браузерах, например Internet Explorer 9.

Обратите внимание на следующее:

- для семейства шрифтов во второй строке показанного выше кода используется общее имя — PTSans, вместо имени, указанного для файла шрифта, — PTSansRegular;
- свойству font-weight присвоено значение normal, поскольку эта версия шрифта не является полужирной (строка 8);
- свойству font-style присвоено значение normal, поскольку это не курсивная версия шрифта (строка 9).

ПРИМЕЧАНИЕ

В приведенных здесь примерах кода предполагается, что файлы шрифтов PTSansRegular.eot, PTSansBold.eot и т. д. находятся в той же папке, что и таблица стилей. Если бы использовались разные папки, пришлось бы изменить URL-адрес для точного указания расположения файлов шрифтов относительно таблицы стилей.

Теперь предположим, что у вас имеется курсивная версия шрифта, у которой имя файла начинается с PTSansItalic. Его следует добавить в таблицу стилей:

```
@font-face {
  font-family: 'PTSans';
  src: url('PTSansItalic.woff2') format('woff2'),
       url('PTSansItalic.woff') format('woff');
  font-weight: normal;
  font-style: italic;
}
```

В строке 2 используется то же самое имя font-family — PTSans. Но значение свойства font-style изменилось на italic (строка 6). Тем самым браузеру сообщается, что указанный шрифт является курсивной версией шрифта PTSans. Нужно также добавить подобные правила @font-face для полужирной, а также полужирной/курсивной версий:

```
@font-face {
  font-family: 'PTSans';
  src: url('PTSansBold.woff2') format('woff2'),
       url('PTSansBold.woff') format('woff');
  font-weight: bold;
  font-style: normal;
}

@font-face {
  font-family: 'PTSans';
  src: url('PTSansBoldItalic.woff2') format('woff2'),
       url('PTSansBoldItalic.woff') format('woff');
  font-weight: bold;
  font-style: italic;
}
```

Иначе говоря, для охвата всех вариантов полужирного, курсивного и обычного начертаний шрифта вам нужны четыре правила @font-face: обратите внимание на то, что во всех случаях используется одно и то же имя шрифта в строке font-family,

а изменяются только значения свойств src (указывающие на разные файлы), font-weight и font-style.

Преимущество этого метода состоит в том, что к тексту можно применить обычный шрифт, добавить в HTML-код элементы em и strong и позволить браузеру взять на себя вопрос выбора файла шрифта для загрузки и использования. В данном примере, если нужно применить шрифт PTSans ко всем абзацам, достаточно добавить следующий стиль в таблицу стилей:

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

Затем нужно разметить абзацы HTML-элементами. Например, так:

```
<p>Когда я был моложе, я мог запомнить <em>все</em>, что было и чего <strong>не</strong>
было. -- <strong><em>Марк Твен</em>
</strong></p>
```

Когда браузер прочитает таблицу стилей (с четырьмя правилами @font-face и стилем элемента p), он отформатирует большую часть абзаца шрифтом PTSansRegular. Для слова «все», заключенного в теги элемента em, будет использоваться шрифт PTSansItalic, а для слова «не» внутри тегов элемента strong — шрифт PTSansBold. Для слов «Марк Твен», окруженных тегами элементов strong и em, будет использоваться шрифт PTSansBoldItalic.

Эти правила работают и для заголовков. Если вы создаете стиль для форматирования всех элементов h1 с использованием шрифта PTSans, то он может иметь такой вид:

```
h1 {
  font-family: PTSans;
}
```

При наличии этого стиля браузер будет фактически использовать полужирную версию шрифта PT-Sans, поскольку заголовки обычно отображаются полужирным шрифтом. (При использовании такой технологии с привлечением нескольких вариантов шрифтов уже не нужно, как раньше, добавлять фрагмент font-weight: normal;.)

Браузер Internet Explorer 8 и его более ранние версии не поддерживают этот метод и станут использовать для всего текста шрифт PTSansRegular. Для элементов em и strong Internet Explorer будет создавать искусственные (псевдо) курсивный и полужирный начертания шрифтов, то есть он начнет наклонять шрифт PTSansRegular на экране для получения курсива и делать шрифт PTSansRegular толще для получения полужирного варианта. Получающиеся в результате сгенерированные компьютером полужирный и курсивный варианты обычно не очень хорошо смотрятся.

Поддержка полужирного и курсивного начертаний в Internet Explorer 8

Если вы все еще учитываете поддержку Internet Explorer 8 (или более раннюю версию), то предыдущее решение по поводу полужирного и курсивного начертаний не будет работать так же хорошо. Поэтому для начала нужно, как и ранее, создать четыре пра-

вила @font-face, по одному для каждого варианта шрифта. Но вместо того, чтобы задавать для них одно и то же имя семейства в свойстве font-family (например, PTSans), каждому из них присваиваем уникальное имя (PTSansRegular, PTSansItalic и т. д.). Иными словами, четыре правила @font-face нужно переписать следующим образом:

```
@font-face { font-family: 'PTSansRegular';
  src: url('PTSansRegular.eot?#iefix') format('embedded-opentype'),
       url('PTSansRegular.woff2') format('woff2'),
       url('PTSansRegular.woff) format('woff');
}

@font-face { font-family: 'PTSansItalic';
  src: url('PTSansItalic.eot?#iefix') format('embedded-opentype'),
       url('PTSansItalic.woff2') format('woff2'),
       url('PTSansItalic.woff) format('woff');
}

@font-face { font-family: 'PTSansBold';
  src: url('PTSansBold.eot?#iefix') format('embedded-opentype'),
       url('PTSansBold.woff2') format('woff2'),
       url('PTSansBold.woff') format('woff');
}

@font-face { font-family: 'PTSansBoldItalic';
  src: url('PTSansBoldItalic.eot?#iefix') format('embedded-opentype'),
       url('PTSansBoldItalic.woff2') format('woff2'),
       url('PTSansBoldItalic.woff') format('woff');
}
```

Обратите внимание, что у каждого правила @font-face имеется собственное название семейства, которое соответствует варианту шрифта: PTSansRegular, PTSansItalic, PTSansBold и PTSansBoldItalic.

Кроме того, обратите внимание на то, что свойства font-weight и font-style, которые использовались ранее, здесь уже отсутствуют за ненадобностью.

Труднее дело будет обстоять с применением шрифта. В предыдущем примере свойство font-family просто применялось к стилю:

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

Теперь вам придется использовать различные имена шрифтов для разных элементов, для p — имя обычного шрифта, для em — курсивного, для strong — полужирного, а чтобы справиться со случаем применения полужирного/курсивного начертания, придется составить селектор потомков. Иначе говоря, чтобы могли работать различные варианты шрифта PTSans, нужно создать четыре стиля, содержащие довольно много строк кода:

```
p {
  font-family: PTSansRegular;
  font-size: 48px;
  font-style: normal;
```

```
  font-weight: normal;
}

p em {
  font-family: PTSansItalic;
  font-style: normal;
  font-weight: normal;
}

p strong {
  font-family: PTSansBold;
  font-style: normal;
  font-weight: normal;
}

p strong em, p em strong {
  font-family: PTSansBoldItalic;
  font-weight: normal;
  font-style: normal;
}
```

В первую очередь обратите внимание на четыре стиля:

- для селектора p используется шрифт PTSansRegular;
- p em представляет собой селектор потомков, используемый для элемента em, находящегося внутри абзаца p, — применяется шрифт PTSansItalic;
- p strong представляет собой еще один селектор потомков, который применяет шрифт PTSansBold к элементу strong внутри абзаца;
- и наконец, мы имеем дело с групповым селектором, составленным из двух селекторов потомков. Первый применяется к элементу em, находящемуся внутри strong в пределах абзаца p. Второй применяется к элементу strong внутри em в пределах абзаца p. Вам нужны оба этих селектора, потому что вы можете вложить элементы em внутрь strong и наоборот.

В результате может получиться такой HTML-код:

```
<p>
  <em><strong>Привет!</strong></em>
  Я говорю с
  <strong><em>тобой</em></strong>
</p>
```

Одиночный селектор p strong em неприменим к слову «Привет!», поскольку оно окружено тегами элемента strong, который находится в элементе em.

ПРИМЕЧАНИЕ

В языке HTML5 элементы b (для полужирного начертания) и i (для курсивного) восстановили свои позиции. Их можно использовать исключительно в оформительских целях, когда нужно, чтобы шрифт текста стал полужирным или курсивным, но без смысловой нагрузки данного форматирования. Например, курсивом часто пишут названия книг, поэтому в данном случае рекомендуется указывать элемент i:

Большая книга CSS

А при использовании элемента em текст будет акцентирован, что заставит программу экранного доступа прочитать его громче, не так, как обычный текст. В любом случае, если вы намерены использовать элементы b и i, не забудьте создать стили для курсивного и полужирного вариантов шрифта (волноваться об этом приходится только в том случае, если речь идет об Internet Explorer 8, для которого нужно обеспечить безопасный способ указания курсива, а первый, рассмотренный ранее, метод таких мер не предусматривает).

Еще нужно обратить внимание на необходимость присваивания во всех этих стилях значения `normal` свойствам `font-weight` и `font-style`. Если этого не сделать, многие браузеры (не только Internet Explorer) будут пытаться утолщать полужирный шрифт и наклонять курсивную версию шрифта.

Эта технология поддержки полужирного и курсивного начертаний, несомненно, требует большого объема работы, который существенно возрастает, если на одном сайте задействовано несколько шрифтов с полужирным и курсивным вариантами. Какой из технологий следует воспользоваться, зависит от того, насколько важна для вас поддержка Internet Explorer 8. На момент написания книги он все еще использовался на 2–11 % компьютеров (по сведениям tinyurl.com/ooevwyr и netmarketshare.com).

ПРИМЕЧАНИЕ

Для решения проблемы поддержки полужирных и курсивных начертаний шрифтов в программе Internet Explorer 8 можно применить и другой подход. Попробуйте воспользоваться первым методом и посмотрите, как выглядят страницы в браузере IE 8. Некоторые шрифты, обычно из числа рубленых, не всегда выглядят так уж плохо, когда IE применяет к ним методы придания псевдополужирного и псевдокурсивного начертания. Может оказаться, что разница не так уж и заметна, и можно воспользоваться первым методом, не испытывая при этом особых проблем. Следует также помнить, что количество устройств, на которых используется IE 8, постоянно снижается по мере приобретения пользователями новых компьютеров, перехода на другие браузеры или же обновления операционных систем.

Использование службы Google Fonts

Если инструкции по использованию веб-шрифтов, рассмотренные в предыдущем разделе, вас смущают, есть более простой способ, хотя и с меньшим выбором шрифтов, предоставленный компанией Google. В дополнение к таким службам, как поиск, карты, электронная почта и множество других, компания Google предоставляет простой в применении сервис веб-шрифтов. Вместо загрузки шрифтов, преобразования их в нужные форматы, а затем размещения на своем веб-сервере вы добавляете одну строку кода со ссылкой на внешнюю таблицу стилей Google, в которой показано, какой шрифт вы предпочитаете использовать. Сервер Google отправляет нужные шрифты в браузер посетителя без всякой путаницы и суеты с вашей стороны.

Вам остается только найти требуемые шрифты на сайте Google Web Fonts, скопировать необходимый код (предоставляемый Google) и добавить его в код своей веб-страницы, после чего создать CSS-стили, использующие указанные шрифты. Сначала нужно посетить сайт Google Fonts, который находится по адресу google.com/fonts (рис. 6.6).

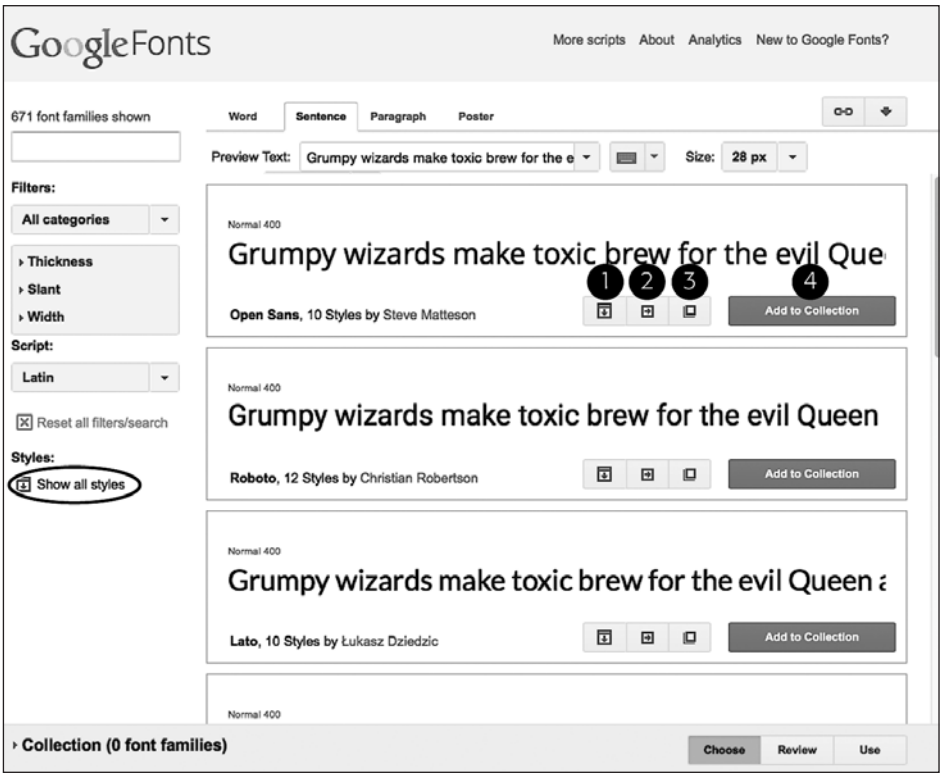

Рис. 6.6. На сайте Google Fonts перечислены шрифты, предлагаемые компанией Google

У некоторых шрифтов есть несколько стилей, например полужирный, курсив, тонкий, ультратонкий и т. д. Для просмотра всех вариантов конкретного шрифта нужно щелкнуть на ссылке Show All Styles (Показать все стили) на боковой панели слева (выделена на рис. 6.6). Кроме того, чтобы увидеть доступные стили конкретного шрифта, нажмите кнопку See all styles (Просмотреть все стили) (см. рис. 6.6, *1*). Чтобы использовать шрифт, нажмите кнопку Quick Use (Быстрое использование) (см. рис. 6.6, *2*), которая вскоре будет рассмотрена. Нажатие кнопки Pop-out (Показать в отдельном окне) (см. рис. 6.6, *3*) приводит к открытию нового окна, в котором представляется подробная информация о шрифте, а также образец, показывающий каждую букву шрифта (хороший способ составить общее представление о внешнем виде алфавита и убедиться в наличии всех необходимых вам символов, например редко используемых или знаков пунктуации). Кнопка Add to Collection (Добавить к коллекции) (см. рис. 6.6, *4*) позволяет добавить шрифт в коллекцию. (Если нужно использовать более одного шрифта, следует все шрифты добавить в коллекцию.)

Поиск и выбор шрифтов

Нужные шрифты выбираются путем создания *коллекции*. Вы находите понравившийся шрифт и нажимаете кнопку Add to Collection (Добавить к коллекции) (см. рис. 6.6).

Чтобы найти шрифт, можно прокрутить главную страницу сайта Google Fonts и просмотреть примеры доступных шрифтов, но при наличии более чем 500 шрифтов поиск нужного шрифта может занять довольно много времени. Если вы уже представляете себе, как должен выглядеть искомый шрифт, например, он должен быть полужирным рубленым шрифтом, предназначенным для заголовков, воспользуйтесь элементами управления в левой части страницы (рис. 6.7).

- **Поиск по имени.** Если известно название интересующего вас шрифта, то его нужно указать в поле поиска (см. рис. 6.7, *1*). К списку шрифтов будет применен фильтр, чтобы показать те из них, которые соответствуют условию поиска.
- **Установка фильтра по категории.** В раскрывающемся списке категорий (см. рис. 6.7, *2*) сбросом флажков можно отфильтровать шрифты, соответствующие одной или нескольким категориям: Serif (Антиквенные), Sans Serif (Рубленые), Display (Экранные), Handwriting (Рукописные) и Monospace (Моноширинные). Чтобы исключить ненужный тип шрифта, следует сбросить соответствующий флажок, а чтобы отобразить — установить. Экранные шрифты обычно бывают полужирными и декоративными, они не подходят для длинных текстовых отрывков, но могут пригодиться для коротких заголовков, акцентирующих внимание на странице. Рукописные шрифты создают иллюзию, что текст написан от руки. Они варьируются от каллиграфических до детских каракулей.
- **Выбор характеристик стиля.** Характеристики шрифтов можно определить с помощью трех ползунков (см. рис. 6.7, *3*). Ползунок насыщенности (Thickness) позволяет подобрать шрифты, контуры которых варьируются от очень тонких (настолько тонких, что надписи трудно читаются, пока не будут отображены шрифтом большого размера) до очень толстых (полужирных и плотных). Ползунок наклона (Slant) фильтрует шрифты по степени наклона: обычно это означает использование курсивных версий шрифтов, но относится и к рукописным шрифтам, которые, как правило, имеют видимый наклон вправо. И наконец, использование ползункового регулятора ширины (Width) позволяет подобрать шрифты зауженного либо расширенного начертания. Используя более широкие шрифты, можно поместить на одну строку всего несколько символов, но зачастую это позволяет придать заголовку более внушительный вид.
- **Выбор набора символов.** И наконец, меню набора символов (Script) (см. рис. 6.7, *4*) позволяет выбрать шрифты с поддержкой символов, отличных от латиницы. Английский язык и многие европейские языки используют набор латинских символов (Latin), но если нужен, например, шрифт для русского текста, можно выбрать набор кириллических символов (Cyrillic). Выберите тот набор символов, который соответствует вашему языку.

Если воспользоваться сразу всеми фильтрами, может не быть никаких результатов. В таком случае следует щелкнуть кнопкой мыши на ссылке Reset all filters/search (Сброс всех фильтров/поиска), выделенной на рис. 6.7, что позволит вернуться к полному списку веб-шрифтов Google Fonts.

ПРИМЕЧАНИЕ

Чтобы посмотреть демонстрацию некоторых лучших шрифтов Google Fonts, перейдите на сайт tinyurl.com/6lz7u7j.

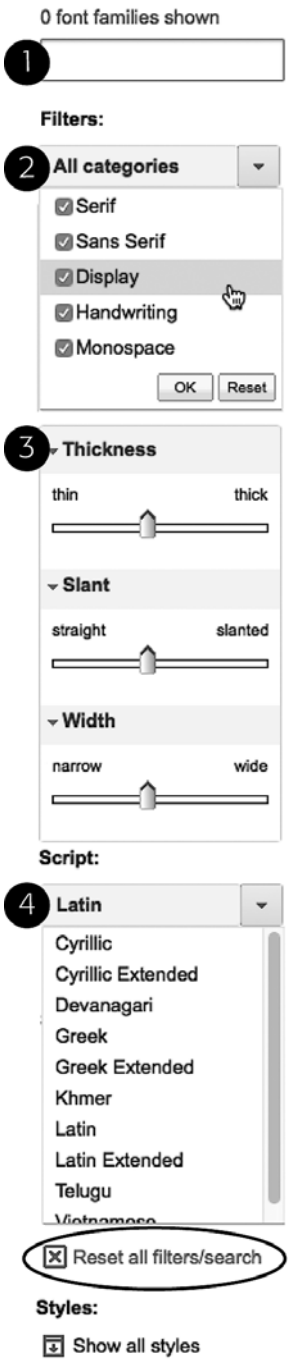

Рис. 6.7. Для содействия поиску шрифтов, соответствующих вашему дизайну, можно провести поиск по каталогу Google Fonts или отфильтровать список шрифтов по различным критериям

Как только нужные шрифты будут найдены, следует нажать кнопку Add to Collection (Добавить к коллекции) (рис. 6.8, *1*). Коллекция представляет собой некое подобие корзины покупателя, в которой можно добавлять и удалять шрифты.

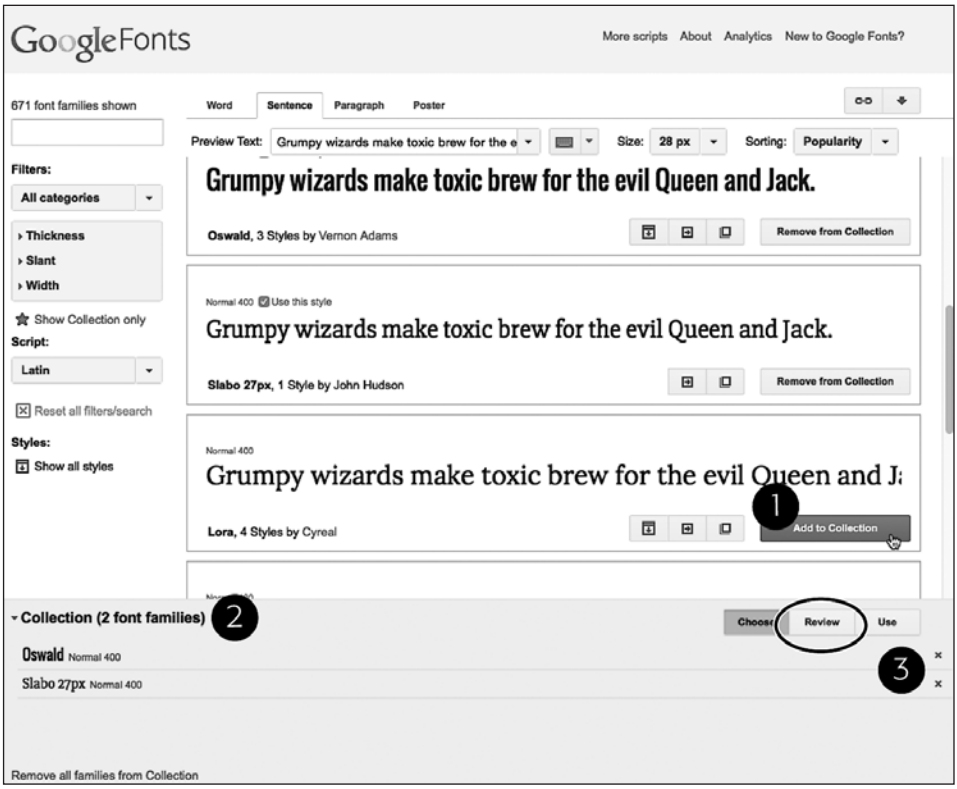

Рис. 6.8. Добавляем шрифты в коллекцию

Чтобы просмотреть шрифты, добавленные в коллекцию, щелкните на стрелке (см. рис. 6.8, *2*). Шрифт можно удалить из коллекции, нажав кнопку с крестиком справа от названия шрифта (см. рис. 6.8, *3*). Кнопка Review (Обзор), выделенная на рис. 6.8, позволяет сравнивать шрифты в коллекции и отображает подробную информацию. Например, можно увидеть полный набор символов для каждого шрифта (то есть каждую букву и символ), протестировать шрифт, печатая собственный текст и изменяя размер шрифта, и даже выполнить сравнение с наложением всех шрифтов в коллекции.

Использование шрифтов Google

После создания коллекции шрифтов настанет черед получения кода, необходимого для их использования.

1. В правом нижнем углу экрана Google Fonts (рис. 6.9, *1*) нажмите кнопку Use (Использовать).

Откроется страница с настройками, а также с кодом, который нужно будет скопировать.

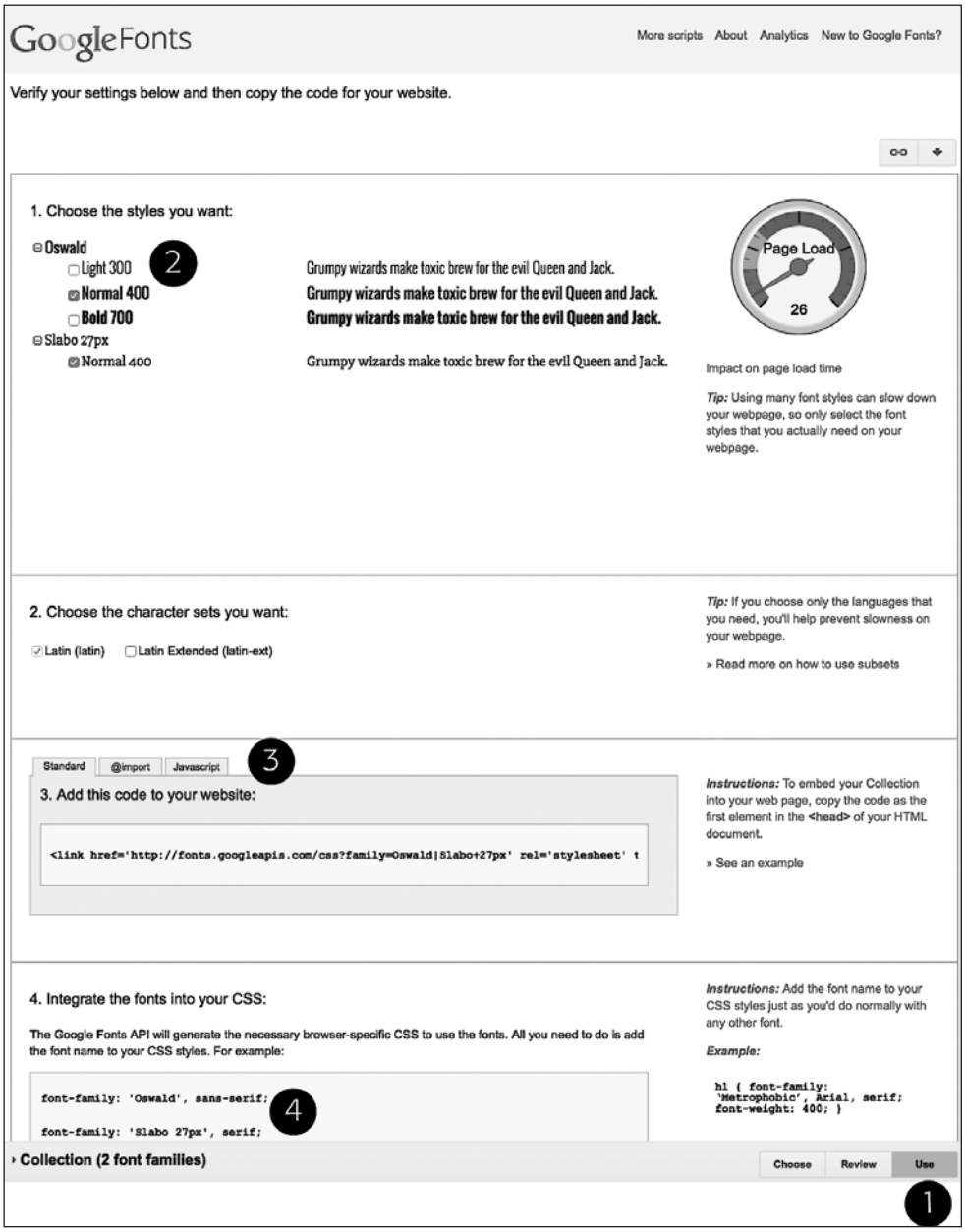

Рис. 6.9. Когда будете готовы к использованию шрифтов, добавленных в вашу коллекцию Google Fonts, нажмите кнопку Use (Использовать) *(1)*, выберите нужный стиль *(2)* и метод, который хотите использовать для добавления этих шрифтов на страницу *(3)*

2. Выберите стиль, который нужно использовать (см. рис. 6.9, *2*).

   Некоторые шрифты включают курсивный, полужирный и другие начертания. Для основного текста страницы обычно нужны как минимум обычный, курсивный и полужирный варианты. Для заголовков можно обойтись и одним вариантом шрифта. Вы, наверное, заметили «обратный спидометр» в правой части страницы. По мере добавления новых стилей и шрифтов его указатель движется по часовой стрелке, показывая, что загрузка шрифтов потребует больше времени.

   Это один из недостатков веб-шрифтов. Поскольку посетители вашего сайта вынуждены их загружать (наряду с веб-страницей, внешними таблицами стилей, графикой и другими элементами, составляющими страницу), нужно поступать разумно и не использовать слишком много шрифтов. В противном случае людям придется долго ждать, пока текст отобразится на экране. Цифры на спидометре показывают среднее количество миллисекунд, затрачиваемых на загрузку файлов шрифтов.

3. Дополнительно выберите тот набор символов, который вам нужен.

   Это действие доступно не для всех шрифтов. Кроме того, при выборе набора символов, отличного от Latin (Латиница) (см. последний пункт предыдущего маркированного списка), можно увидеть и другие варианты, кроме Latin (Латиница) и Latin Extended (Латиница расширенная). Расширенный вариант пригодится в том случае, если ваш текст содержит слова на языке, использующем особые символы, например на турецком, валлийском или венгерском. Для большинства языков с символами латиницы, например французского и испанского, достаточно будет алфавита из набора Latin (Латиница). Если расширенный набор не нужен, то его лучше не использовать, чтобы не увеличивать размер файла и время загрузки шрифта. Для форматирования русскоязычного текста следует выбрать набор Cyrillic (Кириллица). Аналогично латинице набор Cyrillic Extended (Кириллица расширенная) содержит дополнительные редко используемые символы, например церковнославянские буквы.

ПРИМЕЧАНИЕ

Для просмотра перечня дополнительных символов, доступных в наборах Latin Extended (Латиница расширенная) и Cyrillic Extended (Кириллица расширенная), посетите сайт tinyurl.com/p577lm2 и tinyurl.com/ovvavqw соответственно.

4. Скопируйте код в поле Add this code to your website (Добавьте этот код на свой сайт) (см. рис. 6.9, *3*).

   Доступны три вкладки.

   - Standard (Стандарт) — предоставляет элемент `link`, указывающий на внешнюю таблицу стилей (то же самое, что и создание ссылки на любую внешнюю таблицу стилей) на веб-сервере Google и предоставляющий информацию, необходимую Google для доставки нужных шрифтов. Например:

     ```
     <link href='http://fonts.googleapis.com/css?family=Lato:300,400,300italic,
     400italic|Oswald:400,700' rel='stylesheet' type='text/css'>
     ```

     Обратите внимание, что в конце атрибута `href` перечисляются шрифты и их стили. В данном примере используются шрифты Lato и Oswald. Сервер

Google загрузит несколько стилей: 300, 400, 300italic и 700italic для Lato. Эти числа являются способом обозначения веса (насыщенности) шрифта и рассматриваются далее в этом разделе. Кроме того, число и слово italic (например, 300italic) означает шрифт указанной ширины с курсивным начертанием.

ПРИМЕЧАНИЕ

Если сайт Google Fonts предлагает использовать код "type='text/css'" как часть элемента link, ее можно опустить. В версии HTML5 этот код не нужен.

- @import — использование правила @import. Для этого нужно щелкнуть на названной вкладке (см. рис. 6.9, *3*). Преимущество такого подхода заключается в том, что правило @import можно добавить в начало другой таблицы стилей. Предположим, у вас есть одна таблица стилей, предназначенная для вашего сайта, на которую ссылаются все его страницы. Стандартный метод link требует добавления кода к каждой странице сайта. А используя правило @import, можно добавить код к единственной внешней таблице стилей.
- Javascript — подход, требующий использования кода на языке JavaScript. В этой книге данный метод не рассматривается, поскольку он требует большого объема кода, и если вы не сильны в языке JavaScript, то вполне можете допустить ошибку. Кроме того, этот вариант не дает весомых преимуществ по сравнению с двумя другими.

5. Добавьте код на веб-страницы своего сайта.

   Если используется метод link, рассмотренный в предыдущем пункте, то код нужно добавить на каждую страницу, на которой вы собираетесь применять шрифты. Если вы только приступили к процессу создания своего сайта, это сделать нетрудно, но если ваш сайт состоит из большого количества страниц, то придется потрудиться. В таком случае присмотритесь к правилу @import: его можно поместить в верхней части внешней таблицы стилей вашего сайта, после чего все страницы, имеющие ссылку на эту таблицу стилей, также будут использовать нужные шрифты.

ПРИМЕЧАНИЕ

Метод @import может оказать небольшое воздействие на производительность ваших сайтов (то есть немного снизить скорость их работы).

6. Создайте стили, использующие шрифты.

   После того как шрифты загружены, их можно применять практически так же, как и любой другой шрифт. Нужно лишь создать стиль, добавить свойство font-family и указать шрифт. Имя шрифта показано в нижней части страницы сайта Google Fonts (см. рис. 6.9, *4*).

   Если используется несколько начертаний шрифта, нужно также добавить к стилю свойства font-weight и font-style. В Google Fonts обычные ключевые слова normal или bold для обозначения насыщенности шрифтов не указываются. Вместо них задается числовая шкала в диапазоне значений от 100 до 900. Значение 700 соответствует варианту bold, 400 — normal, а остальные числа обозначают

другие варианты толщины. Предположим, вам нужно применить обычную курсивную версию шрифта Gentium Book Basic к элементу `em`. Для этого можно создать следующий стиль:

```
em {
  font-family: "Gentium Book Basic", Palatino, serif;
  font-weight: 400;
  font-style: italic;
}
```

Форматирование текста цветом

Черно-белые оттенки хорошо смотрятся в *классических* и документальных кинолентах, но когда мы имеем дело с текстом, шрифт небесно-синего цвета будет выглядеть намного более изящно и стильно по сравнению с черным. Окрашивание текста веб-страниц средствами каскадных таблиц стилей не представляет трудностей. Фактически мы уже использовали свойство `color` в предыдущих практикумах. Существует несколько способов определения конкретного цвета, но все они базируются на одном принципе: нужно указать само свойство `color` и затем его значение:

```
color: #3E8988;
```

В этом примере значение цвета представлено шестнадцатеричным числом, определяющим слабый оттенок зеленовато-голубого цвета (о *шестнадцатеричных* значениях читайте далее).

В КУРС ДЕЛА!

Adobe TypeKit в качестве альтернативы Google Fonts

Из-за технических и правовых требований к использованию веб-шрифтов некоторые компании взяли на себя все связанные с этим трудности. И Google Fonts не единственный пример. Такие *службы* позволяют выбирать из больших коллекций *шрифтов*, находящихся на собственных веб-серверах компаний. Иными словами, вы не помещаете шрифты на свой сервер, а лишь ссылаетесь на серверы компаний, используя код на языке CSS или JavaScript. Эти службы берут заботу по отправке нужных шрифтов браузерам ваших посетителей на себя (например, EOT для Internet Explorer 8 и более ранних версий).

Коммерческая служба корпорации Adobe, которая называется TypeKit, также предоставляет широкий выбор шрифтов, но за плату. Эта служба является частью корпорации Adobe (которая вдобавок ко всему производимому программному обеспечению занимается также созданием шрифтов) и предоставляет доступ к широкому диапазону профессионально созданных шрифтов. С помощью TypeKit создаются отдельные *наборы* или коллекции шрифтов, которые назначаются сайту. Затем к каждой странице вашего сайта нужно добавить фрагмент JavaScript-кода. Этот код устанавливает подключение к серверам Adobe и доставляет запрошенные вами шрифты посетителям сайта. Adobe TypeKit, помимо платных подписок, предоставляет возможность и бесплатного использования шрифтов, но с ограничениями. В зависимости от количества шрифтов, к которым нужно получить доступ, и ежемесячного количества посетителей вашего сайта абонентская плата может составить от $50 в год. Кроме того, получить доступ к службе можно, оформив подписку Creative Cloud по адресу tinyurl.com/nagp258.

Каждый графический редактор, например Fireworks, Photoshop или GIMP, позволяет указать цвет с помощью шестнадцатеричного или RGB-значения. Кроме того, встроенные в операционные системы Windows и OS X инструменты позволяют определить цвет на цветовом колесе или палитре, а затем преобразовать его в шестнадцатеричное или RGB-значение.

СОВЕТ

Если вам требуется помощь по цветовому оформлению, можете найти множество согласованных коллекций цветов, а также полезных связанных с ними ресурсов во Всемирной паутине по адресу colourlovers.com. По адресу paletton.com находится инструмент для создания веб-цвета и палитры.

Шестнадцатеричные значения цветов

Изначальной системой задания цвета, используемой веб-дизайнерами, является *шестнадцатеричная нотация*. Значение, например `#6600FF`, фактически содержит три шестнадцатеричных числа. В данном примере это `66`, `00` и `FF`. Каждое число определяет уровень красного, зеленого и синего цветов соответственно (как и в цветовой модели RGB, описываемой далее). Окончательное значение цвета получается при смешивании этих трех составляющих.

СОВЕТ

Можете сокращать шестнадцатеричные числа до трех символов, если каждое из трех двухзначных чисел содержит по два одинаковых символа. Например, #6600FF можно привести к виду #60F, а #FFFFFF к #FFF.

RGB

Можно также пользоваться методом перечисления значений по цветовой модели RGB (red — «красный», green — «зеленый», blue — «синий»), с которой вы, возможно, знакомы из графических редакторов. Значение цвета кодируется тремя числами, представляющими процентные соотношения (0–100 %) или числа в диапазоне 0–255 для указания насыщенности каждого компонента (красный, зеленый, синий). Если вы хотите установить белый цвет текста (например, чтобы контрастно выделить его на темном фоне веб-страницы), можно использовать следующее свойство:

```
color: rgb(100%,100%,100%);
```

или

```
color: rgb(255,255,255);
```

СОВЕТ

Вы всегда можете вернуться к классическим именам цветов в языке HTML. Однако это будет минус в новизне и оригинальности вашего сайта. Существует 17 ключевых имен цветов: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, yellow. В CSS-коде их добавляют в стили следующим образом: color: fuchsia; и т. п. Кроме этого, большинство браузеров поддерживает 147 SVG-цветов (которые также называются цветовой схемой X11), поэтому если вам действительно хочется что-то представить в более выгодном цвете, начните использовать такие цвета, как льняной, шоколадный, хаки и белый дым. Перечень имен этих цветов можно найти по адресу tinyurl.com/kqdborf. На странице tinyurl.com/m4jb9e5 имена цветов объединены по оттенкам.

RGBA

Чтобы придать странице выразительность, стоит рассмотреть более современные методы задания цвета. Аббревиатура RGBA означает Red — красный, Green — зеленый, Blue — синий, Alpha — *альфа-канал*, определяющий уровень смешивания цвета с фоном. Этот способ задания цвета работает так же, как и RGB, но с добавлением альфа-канала. То есть можно задать уровень прозрачности, превратив цвет из сплошного в полупрозрачный (рис. 6.10). К значениям цвета по модели RGB добавляется еще одно число с диапазоном от 0 до 1. Значение `0` обозначает полностью прозрачный цвет, а `1` — на 100 % непрозрачный (то есть сквозь него ничего невозможно увидеть):

```
color: rgba(255, 100, 50, .5);
```

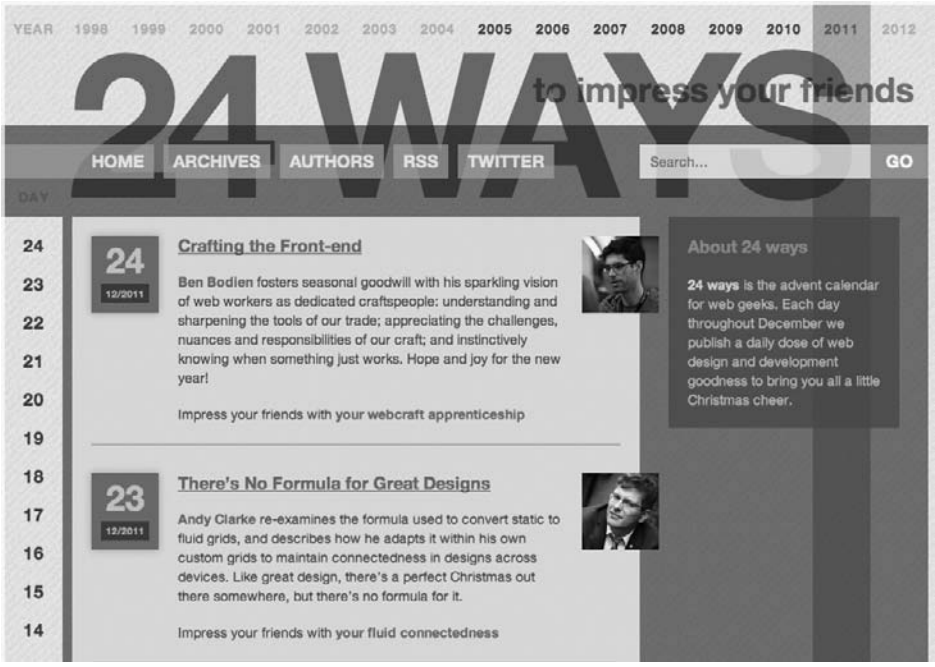

Рис. 6.10. Запись значений по модели RGBA применяется не только для текста. Ею можно пользоваться в сочетании с любым свойством каскадных таблиц стилей, использующим значение цвета, например с цветом фона поля поиска в правом верхнем углу этого изображения или с цветом кнопок навигации: Home, Archives, Authors и т. д.

Накладывая текст, цвет шрифта которого задан с помощью модели RGBA, на фоновые изображения, можно создать интересные визуальные эффекты. Например, можно создавать изображения, частично видимые сквозь символы текста (путем использования такого большого значения, как `.9`) или видимые более явно (значение `.1`).

СОВЕТ

Модель RGBA особенно хорошо поддерживается свойствами text-shadow и box-shadow, которые будут рассмотрены далее. Используя запись значений по модели RGBA, можно создавать более тонкие эффекты отбрасываемых теней, позволяя основной части фона просматриваться сквозь тень.

А в чем же недостатки такого метода? Internet Explorer 8 и его более ранние версии не поддерживают значения по модели RGBA. Один из выходов заключается в задании сначала сплошного цвета с использованием шестнадцатеричной нотации для старых браузеров, а затем второго свойства цвета с помощью модели RGBA:

```
color: rgb(255,100,50); /* для IE8 */
color: rgba(255,100,50,.5); /* для современных браузеров */
```

Первую строку интерпретируют все браузеры, а вторая строка отменяет первую, но только для тех браузеров, которые поддерживают значения по модели RGBA. Иными словами, IE 8 использует первое объявление цвета и игнорирует второе, а IE 9 и другие браузеры используют модель RGBA. Вы просто не увидите в программе IE 8 эффекта прозрачности.

HSL и HSLA

Аббревиатура HSL расшифровывается как Hue — «тон», Saturation — «насыщенность» и Lightness — «светлота» (иногда также используется термин luminance — «яркость»). Это еще один способ задания цвета. Он не поддерживается Internet Explorer 8 и более ранними версиями, но работает во всех остальных браузерах. Если вы используете запись значений по модели RGB или шестнадцатеричные значения, то синтаксис HSL может показаться немного необычным. Вот так выглядит определение ярко-красного цвета:

```
color: hsl(0, 100%, 50%);
```

Внутри конструкции `hsl()` задаются три значения. Первое — значение градуса от `0` до `360` цветового круга. Если вы помните очередность следования цветов в радуге — красный, оранжевый, желтый, зеленый, голубой, синий и фиолетовый, то поймете основную идею значения, через которое выражен цвет. Красный — это `0` (это также и `360`, поскольку это один полный оборот по кругу), желтый — это приблизительно `50`, оранжевый — приблизительно `100`, зеленый — `150` и т. д. Каждый цвет занимает примерно 51°.

Вторым значением указывается насыщенность, или то, насколько чистым является оттенок цвета. Насыщенность указывается в диапазоне значений от `0%` до `100%`. Значение `0%` означает полное отсутствие насыщенности, или тусклый серый оттенок. Фактически неважно, какой тон задан, `0%` всегда даст один и тот же серый тон. Значение `100%` задает чистый цвет, яркий и живой. Третье значение определяет степень светлоты, указанную в процентах от `0%` (полностью черный) до `100%` (полностью белый). Если нужно получить чистый цвет, следует воспользоваться значением `50%`.

Предполагалось, что метод HSL будет интуитивно понятнее, чем RGB или шестнадцатеричные значения, но если вы считаете, что его трудно освоить, можете им не пользоваться. Вместо этого обратитесь к Fireworks или Photoshop либо воспользуйтесь палитрой цветов во Всемирной паутине, что существенно упростит выбор значения по модели RGB или шестнадцатеричного значения.

ПРИМЕЧАНИЕ

Если вас заинтересовала модель HSL, то по адресу hslpicker.com можно найти простую в применении палитру цветов.

Точно так же, как у RGB есть сопутствующий формат RGBA с альфа-каналом, HSL поддерживает прозрачность с помощью формата HSLA. Он работает аналогично RGBA. Значение прозрачности указывается в диапазоне от `0` (невидимый) до `1` (полностью непрозрачный). Следующее свойство приведет к созданию ярко-красного цвета с 50%-ной прозрачностью:

```
color: hsla(0, 100%, 50%, .5);
```

Примеры в книге больше ориентированы на применение моделей RGB и RGBA, но если HSL вам более понятен, используйте этот формат!

Изменение размера шрифта

Установка определенного размера шрифта текста веб-страницы — хороший способ визуально придать тексту значимость и привлечь внимание посетителей к наиболее важным фрагментам страницы. Заголовки, отображенные шрифтом большего размера, привлекают внимание. В то же время информация, отображенная маленьким, возможно, подстрочным шрифтом, позволяет этому блоку не выделяться на фоне общего текста веб-страницы, создавая ненавязчивый комментарий.

Свойство `font-size` устанавливает размер шрифта текста. За значением всегда должна следовать единица измерения величины. Например, так:

```
font-size: 1em;
```

Сочетание значения свойства и единицы измерения, которые вы указываете для установки размера шрифта (в данном примере `1em`), определяют размер шрифта текста. Каскадные таблицы стилей предлагают большой выбор единиц измерения: предопределенные ключевые слова, em (стандартная единица измерения в типографской системе: размер буквы M, напечатанной шрифтом Cicero), ex (то же самое, только берется размер буквы X), пикселы, проценты, пики, пункты, дюймы, сантиметры и миллиметры.

Единицы измерения, обычно используемые в печати (пики, пункты, дюймы и т. д.), не очень удобно применять к веб-страницам, так как невозможно предугадать их вид на разных мониторах. Однако пункты удобно использовать при создании таблиц стилей для веб-страниц, ориентированных для печати на принтере. Только небольшую часть всех существующих единиц измерения — пикселы, ключевые слова, em, проценты — можно применять для определения размеров шрифтов текста веб-страниц, отображаемых браузером на экране монитора. В следующей части книги рассказывается, как они работают.

Пикселы

Значения в пикселах наиболее просты для понимания, поскольку не зависят от настроек браузера. Когда вы определяете, например, размер шрифта равным 36 пикселов для заголовка `h1`, браузер отображает текст высотой 36 пикселов. Дизайнеры выбирают эти единицы измерения потому, что они обеспечивают

постоянные согласованные параметры размеров текста на различных типах компьютеров и браузеров.

Чтобы присвоить свойству `font-size` значение в пикселах, введите число с символами `px`:

```
font-size: 36px;
```

ПРИМЕЧАНИЕ

Не добавляйте пробел между значением свойства и единицей измерения. Например, правильно 36px, но не 36 px.

В КУРС ДЕЛА!

Пикселы и Retina-дисплеи

Когда компания Apple представила первые плееры iPod и смартфоны iPhone с Retina-дисплеем (начиная с модели iPhone 4 и iPod 4-го поколения), владельцы этих устройств радовались яркости и четкости изображений. Retina-дисплеи компании Apple предоставляют более четкое изображение за счет использования большего количества пикселов на квадратный дюйм. В то время как в обычных компьютерных дисплеях этот показатель варьируется в пределах 72–100 пикселов на дюйм, Retina-дисплеи могут похвастаться значением вплоть до 400 пикселов на дюйм.

С тех пор Apple использует Retina-дисплеи (за исключением плееров и смартфонов) в таких устройствах, как планшеты iPad и компьютеры iMac/MacBook. Другие изготовители планшетов и компьютеров пытаются не отставать, предлагая устройства с дисплеями с существенно большим количеством пикселов на дюйм, чем ранее. Что все это означает для веб-дизайнеров? Существенное увеличение объема работы. Далее мы еще вернемся к этой теме. Из врезки «Когда пиксел уже не пиксел?» главы 15 вы узнаете, что Retina-дисплеи существенно влияют на изображения и вам придется потрудиться, чтобы изображения на таких экранах выглядели достойно.

Что же касается пикселов, упомянутых выше, то браузеры устройств, оснащенных Retina-дисплеями, фактически умножают значение в пикселах на 2. Иначе говоря, если указать для текста размер 16 пикселов, то браузеры, подстраиваемые под Retina-дисплей, фактически будут использовать на экране для прорисовки текста 32 пиксела. Это хорошая новость. Если бы браузер для отображения текста использовал только 16 из своих маленьких пикселов, то никто не смог бы прочитать текст на Retina-дисплее.

Ключевые слова, проценты и единица измерения em

Все перечисленные далее способы изменения размера шрифта средствами CSS с помощью ключевых слов, процентных значений и единиц em влияют на текст, отображаемый в окне браузера, в соответствии с некоторыми правилами. Другими словами, если вы не определите размер средствами каскадных таблиц стилей, то браузер применит к тексту свои предопределенные параметры. В большинстве случаев обычный текст, находящийся вне элементов заголовков, отобразится высотой 16 пикселов — это *базовый размер шрифта* текста.

Пользователи могут корректировать настройки браузеров, увеличивая или уменьшая базовый размер шрифта, но для этого нужно ковыряться в исходных настройках браузера, с чем большинство людей не захочет связываться.

ПРИМЕЧАНИЕ

У большинства браузеров имеется функция масштабирования, позволяющая увеличить/уменьшить весь контент на странице. Выбор масштаба на самом деле не изменяет базовый размер шрифта текста. Нажатие сочетания клавиш Ctrl++ (⌘++) на большинстве браузеров приводит к увеличению, а сочетания Ctrl+– (⌘+–) — к уменьшению масштаба. Удерживание нажатой клавиши Ctrl (^) и использование колесика мыши также позволяет увеличивать и уменьшать масштаб страницы.

Когда вы изменяете текст с помощью каскадных таблиц стилей, браузер берет за основу базовый размер шрифта текста (будь то первоначальный высотой 16 пикселов или другой) и корректирует его в большую или меньшую сторону в соответствии со значением ключевого слова, единицей измерения em или процентным отношением.

Ключевые слова

Каскадные таблицы стилей предлагают семь ключевых слов, которые позволяют назначить размер шрифта относительно базового: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large` и `xx-large`. CSS-код будет выглядеть следующим образом:

```
font-size: large;
```

Среднее значение `medium` — базовый размер шрифта браузера. Остальные значения уменьшают или увеличивают размер шрифта с различными коэффициентами. Другими словами, несмотря на то, что, казалось бы, каждое изменение размера должно быть последовательным увеличением или уменьшением предыдущего значения, это не так. Обычно `xx-small` является эквивалентом 9 пикселов (принимая во внимание, что вы не изменяли базовый размер шрифта в своем браузере), `x-small` соответствует 10 пикселам, `small` — 13, `large` — 18, `x-large` — 24, а `xx-large` — 32 пикселам.

Как вы заметили, имеется ограниченный набор ключевых слов — всего семь вариантов. Если требуется большая свобода действий по масштабированию, надо обратиться к другим средствам и единицам измерения, описываемым далее.

Процентные значения

Подобно ключевым словам, процентные значения изменяют размер текста относительно базового размера шрифта, определенного браузером. Они обеспечивают более гибкое и точное управление, чем ключевые слова `large`, `x-large` и т. д. Любой браузер имеет предопределенный базовый размер шрифта. У большинства он составляет 16 пикселов. Можно изменить это значение в настройках браузера. Независимо от настройки, основной размер шрифта эквивалентен 100 %. Другими словами, по умолчанию он равнозначен установке шрифта высотой 16 пикселов.

Допустим, вы хотите отобразить определенный заголовок в два раза крупнее, чем средний шрифт веб-страницы. Для этого установите размер шрифта, равный 200 %:

```
font-size: 200%;
```

Или, если хотите, чтобы шрифт был немного меньше базового, укажите значение 90 %.

Приведенные примеры являются самыми простыми, но на практике встречаются более сложные ситуации. Например, относительный размер 100 % может изменяться, если наследуется значение элемента-предка.

В левом нижнем углу веб-страницы, показанной на рис. 6.11, есть текст в элементе `div`, шрифт которого установлен размером 200 %. Это в два раза больше базового размера и составляет 32 пиксела. Все вложенные элементы наследуют его и используют для вычисления собственных размеров шрифтов. Другими словами, для элементов, вложенных в `div`, размер 100 % равен 32 пикселам. Так, текст заголовка `h1`, находящегося внутри элемента `div`, составляет 100 % и отображается в два раза больше базового для этой веб-страницы, то есть высотой 32 пиксела.

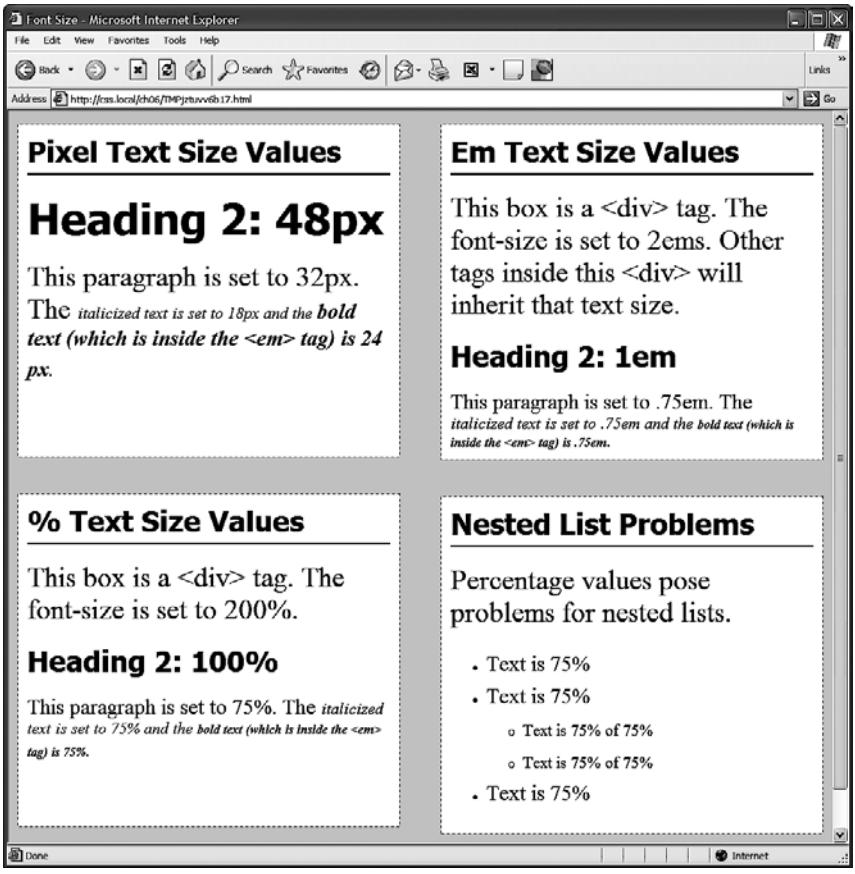

Рис. 6.11. Пикселы, em и проценты — три самые распространенные и наиболее применяемые единицы измерения для установки размеров шрифтов. Будьте внимательны при использовании единиц em и процентных значений, когда есть наследуемые свойства размеров шрифтов. Если вы заметили, что какой-либо текст выглядит не так, как предусмотрено, а необычно крупным или мелким, убедитесь в том, что он не наследует параметры. Или попробуйте использовать значение rem

Единица измерения em

Если вы поняли, как вычисляются процентные значения, то легко поймете и единицу em. Принцип их применения практически одинаков, но большинство веб-дизайнеров все же используют em, так как она применяется в книгопечатании.

Слово *em* заимствовано из типографского дела. Оно имеет отношение к размеру заглавной буквы «M» определенного шрифта. В мире веб-дизайна это слово приобрело собственное значение. В каскадных таблицах стилей понятие относится к базовому размеру шрифта текста. Иными словами, значение размера шрифта `1em` означает то же самое, что и 100 %, как описано в предыдущем разделе. Иначе говоря, процентное значение эквивалентно em, умноженному на 100: `.5em` — 50 % и т. д.

Например, этот CSS-код задает то же форматирование, что и `font-size: 200%;`:

```
font-size: 2em;
```

ПРИМЕЧАНИЕ

Как и при использовании пикселов, вы не должны указывать пробел между числом и единицей измерения em.

Принцип работы механизма наследования с em точно такой же, как и с процентами. Взгляните, например, на верхний правый угол рис. 6.11. Шрифт нижнего абзаца установлен равным `.75em`, а поскольку элемент `p` наследует размер шрифта `2em` (32 пиксела) от контейнера `div`, в результате получается размер шрифта: 0,75 × 32 = 24 пиксела. Внутри абзаца `p` есть два других элемента, свойству `font-size` которых присвоено значение `.75em`. Размер шрифта элемента наиболее глубокой вложенности `strong` установлен равным `.75em`, или, по сути, 75 % от *унаследованного*. Довольно сложный расчет: 32 пиксела (размер, унаследованный от `div`) × 75 (размер от `p`) × 0,75 (размер `em`) × 75 (собственный размер `strong`). В результате вычислений получается приблизительно 14 пикселов.

Унаследованные значения свойства `font-size` могут вызвать проблемы для вложенных (многоуровневых) списков (см. рис. 6.11). Например, у вас имеется стиль `ul { font-size: 75% }`, а затем вы создаете вложенный список, то есть внутри `ul` расположены другие элементы. Получается, что для вложенного `ul` должен быть установлен размер шрифта, равный 75 % от внешнего `ul`. Следовательно, подпункты списков будут отображены меньшим шрифтом, чем пункты, и так далее применительно к подпунктам следующего подуровня.

Чтобы обойти эту проблему, создайте дополнительный стиль с селектором потомков (см. раздел «Форматирование вложенных элементов» главы 3): `ul ul {font-size: 100%}`. Этот стиль устанавливает размер шрифта любого элемента `ul`, вложенного в другой `ul`, равным 100 %. Другими словами, 100 % от размера шрифта родительского элемента `ul`, в который вложен другой элемент. В данном примере это обеспечивает сохранение размера шрифта вложенных подпунктов списков равным 75 % от базового размера. Это еще один способ предотвратить эффект изменения шрифта вследствие наследования.

Единица измерения rem

В спецификацию CSS3 включена новая единица измерения под названием rem. Это название означает *Root EM*, то есть его значение основано на размере текста корневого (root) элемента. В большинстве случаев это относится к базовому размеру шрифта текста, поэтому значение `.75em`, показанное на рис. 6.11, можно заменить на такое:

```
font-size: .75rem;
```

Этот стиль задает размер шрифта равным 0,75 единиц от базового, а не текущего размера шрифта (как в случае применения единиц измерения em). Корневым элементом в языке HTML считается `html`, который можно найти в начале веб-страницы. При использовании значений rem можно установить базовый размер шрифта текста элемента `html`, а затем использовать единицу измерения rem для форматирования текста относительно этого базового размера. Например, можно установить базовый размер шрифта текста равным 20 пикселам:

```
html {
  font-size: 20px;
}
```

А затем воспользоваться единицей измерения rem для создания стиля относительно этого 20-пиксельного базового размера шрифта текста. Тогда, чтобы задать для всех абзацев размер шрифта текста, равный 15 пикселам, нужно добавить такой стиль:

```
p {
  font-size: .75rem;
}
```

Единица измерения rem позволяет избежать проблем наследования размера шрифта, с которыми сталкиваются дизайнеры при использовании процентных значений и единиц `em`. Процентные значения и единицы em основаны на размере шрифта их родительского элемента. Размер шрифта, заданный в процентном значении и примененный к нескольким вложенным элементам, приведет к усложнению, изменяя каждый последующий размер шрифта на определенный процент, по отношению к предыдущему. Тем не менее единицы rem всегда основаны на размере шрифта элемента `html`. Другими словами, они всегда сохраняют один и тот же размер шрифта, даже если вложены в элементы, которые наследуют различное форматирование.

Но нужно иметь в виду, что такую единицу, как rem, поддерживают на данный момент все основные браузеры, кроме Internet Explorer 8 и его более ранних версий.

СОВЕТ

Текст веб-страницы выделяют различными способами. Этого можно добиться, изменив размер шрифта определенного фрагмента текста, отдельных слов или с помощью контрастности. Окрашивание текста в темный, светлый или более яркий оттенок визуально выделяет его. Применение контрастности — одно из наиболее важных правил хорошего графического дизайна. Контрастность может помочь выделить важные сообщения, фрагменты текста, привлечь внимание. Краткий обзор особенностей управления контрастностью в типографике описан во Всемирной паутине по адресу tinyurl.com/ng39zb9.

Форматирование символов и слов

Вам потребуется немало времени для точной и тонкой настройки параметров текста: цвета, размеров шрифтов и т. д. Однако каскадные таблицы стилей предоставляют также множество других средств для форматирования текста. Из самых

распространенных свойств можно выделить, например, полужирное и курсивное начертания, а из менее широко используемых — создание капители, изменение межбуквенного интервала и т. д.

СОВЕТ

Средства языка CSS позволяют комбинировать множество свойств форматирования текста. Излишне «тяжеловесное» оформление усложняет страницу для чтения и понимания. Хуже всего, если из-за такого форматирования остается негативное впечатление от посещения сайта.

Курсивное и полужирное начертания

Браузеры отображают текст, заключенный внутри элементов `em` и `i`, *курсивом* (шрифтом с наклонным начертанием), а текст, находящийся в элементах `strong`, `b`, `th` (table header — «заголовок таблицы»), заголовки (`h1`, `h2` и т. д.) — **полужирным**. Вы можете управлять этим форматированием. Можно отменить полужирное начертание для заголовков или выделить курсивом обычный текст, используя свойства `font-style` и `font-weight`.

Чтобы выделить текст курсивом, добавьте к стилю следующий код:

```
font-style: italic;
```

Или, наоборот, можете установить для текста обычный, *не* курсивный шрифт:

```
font-style: normal;
```

ПРИМЕЧАНИЕ

Свойство font-style также содержит третью команду oblique, которая имеет тот же эффект, что italic.

Свойство `font-weight`, определяющее вес (насыщенность) шрифта, позволяет форматировать текст полужирным или обычным начертанием. Фактически, согласно правилам каскадных таблиц стилей, можно указать любое из девяти числовых значений (в диапазоне от 100 до 900) для определения точных, едва различимых градаций насыщенности (от «супернасыщенного» (900) до «крайне легкого, почти невидимого» веса 100) шрифта. Естественно, чтобы эти команды работали, сами используемые шрифты должны иметь девять различных значений насыщенности, обеспечивая тем самым визуальный эффект. Единственный способ использования числовых значений касается рассмотренных ранее веб-шрифтов. Фактически числовые значения применяются службой Google Fonts исключительно для задания насыщенности шрифта.

ПРИМЕЧАНИЕ

При использовании веб-шрифтов форматирование текста полужирным и курсивным начертанием требует выполнения ряда других, рассмотренных ранее в этой главе действий.

Чтобы сделать шрифт текста полужирным, используйте такой код:

```
font-weight: bold;
```

Обычный шрифт устанавливается следующим образом:

```
font-weight: normal;
```

СОВЕТ

Поскольку для заголовков уже предопределен стиль с полужирным начертанием, возможно, потребуется искать другой способ для выделения, которое вы хотите придать некоторым словам заголовка. Вот один из способов:

```
h1, strong {color: #3399FF; }
```

Этот стиль, с селектором потомков, меняет цвет всех элементов strong (обычно отображаемых полужирным шрифтом), заключенных внутри заголовка h1.

Прописные буквы

Набрать текст прописными буквами очень просто: нажмите клавишу Caps Lock и вводите текст. Что же делать, если нужно, чтобы прописными буквами были выделены все уже набранные заголовки веб-страницы или текст со строчными буквами, который скопировали и вставили из редактора Microsoft Word? Вместо того чтобы повторно набирать текст заголовков, обратитесь к свойству `text-transform`. С его помощью можно преобразовать любой текст в верхний или нижний регистры или даже превратить первые буквы каждого слова в прописные (для названий и заголовков). Ниже представлен пример преобразования в прописные буквы:

```
text-transform: uppercase;
```

Сделать буквы строчными можно с помощью значения `lowercase`, а превратить первые буквы в прописные — с помощью слова `capitalize`.

Поскольку это свойство наследуемо, любые элементы, вложенные в теги со свойством `text-transform`, приобретают то же форматирование: верхний регистр, нижний, прописные буквы. Чтобы *запретить* изменять регистр текста, используйте значение свойства `none`:

```
text-transform: none;
```

Капитель. Для придания тексту типографической изысканности можно использовать свойство `font-variant`, которое позволяет преобразовать текст таким образом, что все буквы станут капителью (малыми прописными). В этом стиле строчные буквы выглядят как немного уменьшенные прописные. Большие фрагменты такого текста становятся трудночитаемыми, но стиль придает веб-странице старосветскую, книжную многозначительность. Для создания стиля с малыми прописными буквами используйте следующий код:

```
font-variant: small-caps;
```

Декорирование текста

Каскадные таблицы стилей позволяют использовать также свойство `text-decoration` для улучшения внешнего вида текста путем добавления различных дополнительных элементов. С его помощью можно добавить линии подчеркивания, надчеркивания, зачеркнуть текст (рис. 6.12) или сделать его мигающим. Свойство `text-decoration` применяется в сочетании со следующими ключевыми словами: `underline`, `overline`, `line-through` или `blink`. Например, чтобы подчеркнуть фрагмент, наберите код:

```
text-decoration: underline;
```

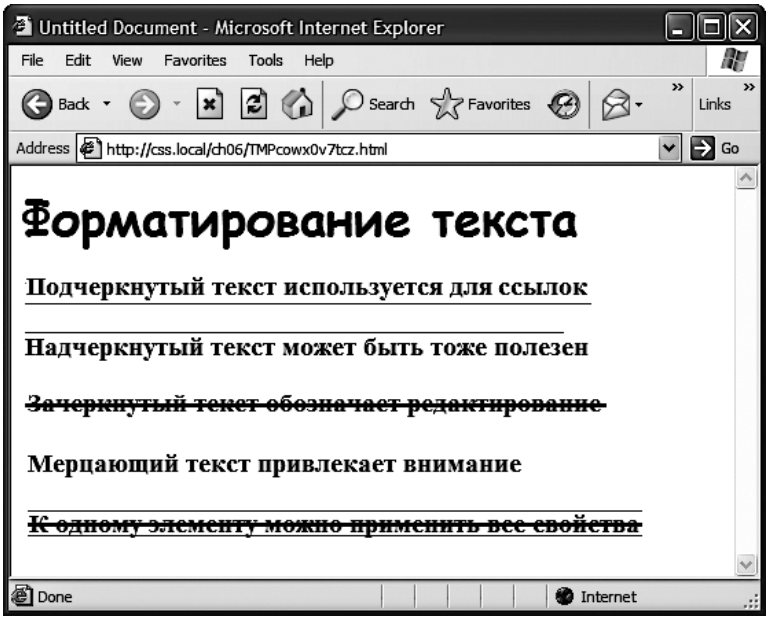

Рис. 6.12. Результат применения свойства text-decoration

Вы можете также объединять несколько ключевых слов вместе для комбинирования эффектов. Добавить к тексту линии подчеркивания и надчеркивания одновременно можно следующим способом:

```
text-decoration: underline overline;
```

Однако не стоит изощряться в таком «вычурном» форматировании только потому, что *есть такая возможность*. Во-первых, у всех, кто посещает Всемирную паутину на протяжении долгого времени, любой подчеркнутый текст инстинктивно ассоциируется с гиперссылкой, возникает желание непременно щелкнуть на ней. Таким образом, подчеркивать слова, не являющиеся частью гиперссылки, будет не очень хорошей идеей. Установив значение мерцания `blink`, вы на самом деле сделаете текст больше походящим на неоновую вывеску типа «Для любителей!» (к тому же большинство браузеров не сделают текст мерцающим, даже если вы укажете это свойство).

ПРИМЕЧАНИЕ

Вы можете применить эффект, подобный подчеркиванию и надчеркиванию, если добавите к элементу — в данном случае тексту — линию границы border сверху или снизу (см. раздел «Добавление границ» главы 7). Преимущество состоит в том, что с помощью свойства border вы можете управлять одновременно многими параметрами: размещением, размером, цветом линий-рамок и в конечном счете придать тексту более привлекательный вид, не делая его похожим на гиперссылки.

Значение `overline` свойства `text-decoration` помещает линию над текстом, а `line-through` зачеркивает текст. Некоторые веб-дизайнеры применяют последний эффект, чтобы показать читателю, что фрагмент был удален из первоначального варианта документа.

Отменить декорирование полностью можно с помощью ключевого слова none:

```
text-decoration: none;
```

Зачем вам может понадобиться отмена декорирования? Самый распространенный пример — удаление стандартной подчеркивающей линии у гиперссылок (см. раздел «Форматирование ссылок» главы 9).

Интервал между символами и словами

Другой способ выделения текстового фрагмента состоит в регулировании интервала, с которым отображаются отдельные буквы или слова (рис. 6.13). Уменьшение межсимвольного интервала (трекинга) с помощью свойства letter-spacing поможет сжать текст заголовков. Они будут казаться еще более плотными и визуально «тяжелыми», а заодно вмещать больше текста на единственной строке заголовка. И наоборот, увеличение интервала может придать заголовкам более спокойный, величественный вид. Чтобы это сделать, используйте отрицательные значения свойства:

```
letter-spacing: -1px;
```

Положительные значения свойства делают промежуток между буквами больше:

```
letter-spacing: .7em;
```

Аналогично можно изменить интервал между словами, используя свойство word-spacing. Оно увеличивает/уменьшает промежуток между словами, не затрагивая интервалы между буквами внутри слов:

```
word-spacing: 2px;
```

С этими свойствами можно использовать любые единицы измерений, применимые к тексту: пикселы, em, проценты (с положительными или отрицательными значениями).

Если вы хотите, чтобы текст был удобен для чтения, применяйте небольшие значения интервалов. В противном случае буквы и слова будут перекрываться, накладываться друг на друга. Чтобы обеспечить комфортное представление текстового контента сайта, аккуратно настраивайте интервалы между символами и словами.

Добавление тени

В спецификации CSS3 включено одно свойство, позволяющее добавлять к тексту тени для придания глубины и выразительности заголовкам, спискам и абзацам (рис. 6.14).

Свойство text-shadow требует задания трех параметров: горизонтального смещения (насколько левее или правее текста должна отображаться тень), вертикальное смещение (насколько выше или ниже текста должна появиться тень), степень размытости тени и цвет отбрасываемой тени. Например, эффект, показанный в верхней части рис. 6.14, задается следующим кодом:

```
text-shadow: -4px 4px 3px #999999;
```

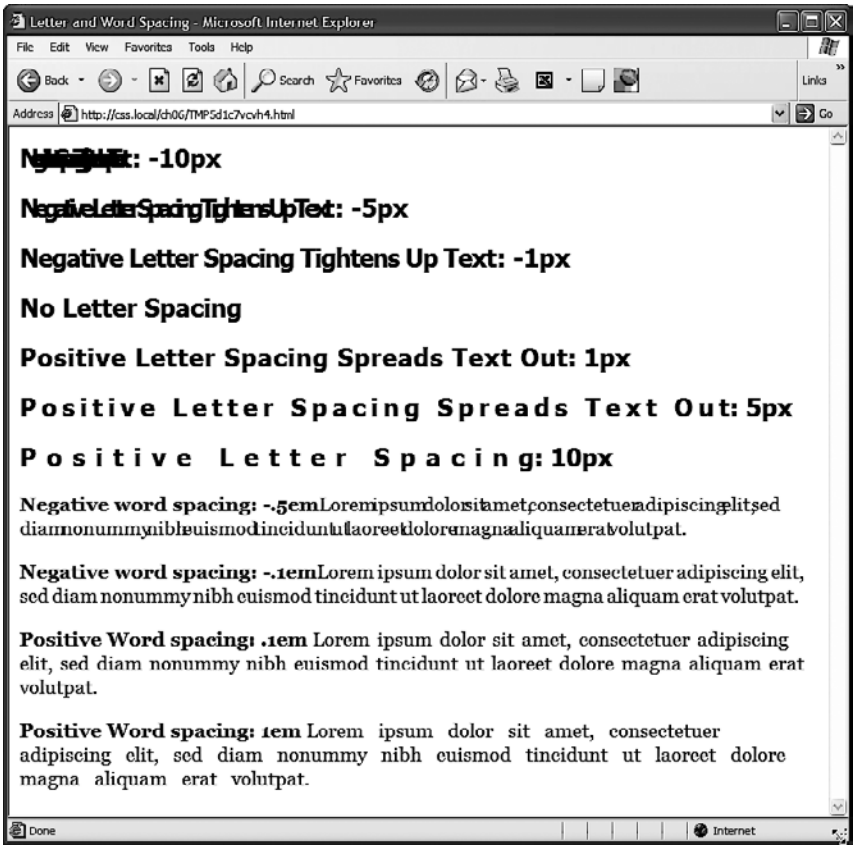

Рис. 6.13. Слишком большое или слишком маленькое значение интервала между символами либо словами может ухудшить читабельность текста

Первое значение, `-4px`, говорит «отобразить тень на 4 пиксела левее текста» (положительное значение приведет к отображению тени правее текста). Второе значение, `4px`, задает отображение тени на 4 пиксела ниже текста (отрицательное значение приведет к отображению тени над текстом). Значение `3px` определяет степень размытости тени (значение `0px` (без размытости) приводит к отбрасыванию четкой тени, и чем больше будет значение, тем более размытой будет тень). И наконец, последнее значение определяет цвет отбрасываемой тени.

Для создания более сложных эффектов можно даже добавить несколько отбрасываемых теней (см. рис. 6.14, *внизу*): нужно добавить запятую, а после нее — дополнительные значения отбрасываемой тени:

```
text-shadow: -4px 4px 3px #666, 1px -1px 2px #000;
```

Количество добавляемых таким образом теней ничем не ограничивается (кроме вашего вкуса). Этот эффект не поддерживается в Internet Explorer 9 и более ранних версиях браузера. Но он работает во всех других современных браузерах (даже в более поздние версии Internet Explorer). Иными словами, чтобы текст лучше

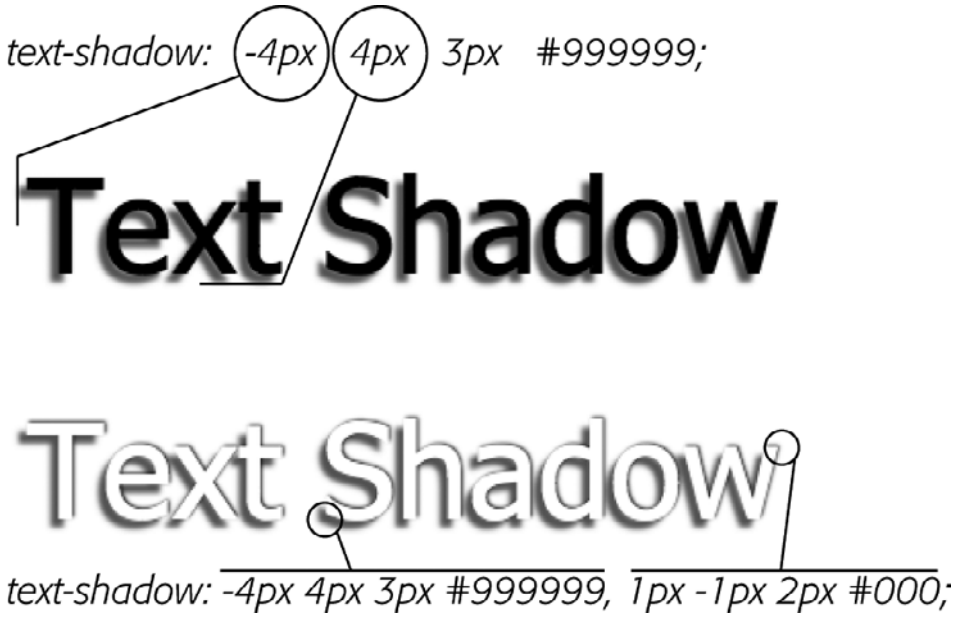

Рис. 6.14. Текст с тенью — отличный способ придания едва заметной (но если вы на этом не настаиваете, то и весьма заметной) глубины заголовкам и прочему контенту. Но свойство text-shadow не поддерживается в Internet Explorer 9 и более ранних версиях браузера

читался, полагаться на этот эффект *не* стоит. Изображение в нижней части рис. 6.14 показывает, что не нужно делать: использовать белый цвет текста, обеспечивая его читаемость исключительно за счет отбрасываемой тени, определяющей очертания текста. В Internet Explorer 9 и более ранних версиях браузера белый текст на белом фоне виден не будет.

ПРИМЕЧАНИЕ

Чтобы увидеть красивые приемы использования текстовых теней, посетите сайт по адресу tinyurl.com/kh5dj2s. Прекрасный пример создания трехмерного текста с помощью нескольких текстовых теней находится по адресу tinyurl.com/kh5dj2s.

Форматирование абзацев

В CSS есть свойства, которые используются для форматирования не отдельно взятых слов, а фрагментов, блоков текста. Иначе говоря, их можно применять к целым абзацам, заголовкам и т. д.

Установка межстрочного интервала

В дополнение к настройке интервала между словами и символами каскадные таблицы стилей позволяют устанавливать межстрочный интервал *(интерлиньяж)* — промежуток между базовыми линиями двух соседних строк текста, используя свойство

line-height. Чем больше межстрочный интервал, тем больше промежуток между отдельными строками (рис. 6.15).

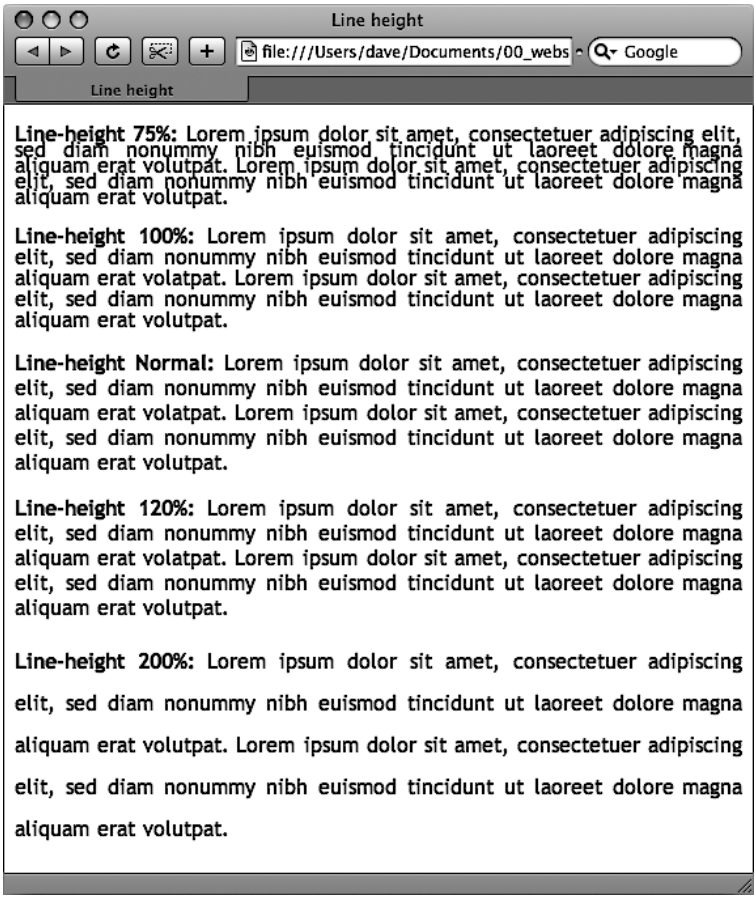

Рис. 6.15. Свойство установки межстрочного интервала, line-height, позволяет растянуть строки абзаца или расположить их ближе друг к другу, с меньшим промежутком. Стандартный межстрочный интервал эквивалентен 120 %. Таким образом, *вверху* мы видим, что меньшее процентное значение сжимает строки, а большее значение — раздвигает *(внизу)*

Интерлиньяж в пикселах, единицах em и процентах

Как и в свойстве размера шрифта, font-size, вы можете использовать пикселы, единицы em или проценты для установки размера межстрочного интервала с помощью свойства line-height:

```
line-height: 150%;
```

Вообще, процентные значения или единицы em лучше, нежели пикселы: размер, установленный в этих единицах измерения, напрямую зависит от параметров шрифта и автоматически корректируется пропорционально изменению значения свойства font-size. Так, если вы установите межстрочный интервал равным 10 пикселов

и затем измените на гораздо больший (например, 36 пикселов), то высота останется равной 10, а строки будут накладываться друг на друга. Однако при использовании значения размера в процентах (скажем, 150 %) межстрочный интервал корректируется пропорционально всякий раз, когда вы изменяете размер шрифта в соответствующем свойстве `font-size`.

Стандартный размер межстрочного интервала браузера составляет 120 %. Таким образом, когда вы хотите уменьшить высоту строк, промежуток между ними, используйте меньшее значение. Соответственно, чтобы увеличить межстрочный интервал, то есть распределить строки дальше друг от друга, используйте значение больше 120 %.

ПРИМЕЧАНИЕ

Чтобы определить размер межстрочного интервала, браузер вычитает высоту шрифта из высоты строки. В результате получается значение интерлиньяжа между двумя соседними строками текста абзаца. Допустим, размер шрифта составляет 12 пикселов. Межстрочный интервал, установленный в размере 150 %, в итоге равняется 18 пикселам. Таким образом, браузер добавляет пустой промежуток размером 6 (18 — 12) пикселов между двумя строками текста.

Интерлиньяж в виде числового значения

Каскадные таблицы стилей предлагают еще одну единицу измерения для установки размера межстрочного интервала — обычное числовое значение. CSS-код выглядит следующим образом:

```
line-height: 1.5;
```

После этого значения не нужно указывать единицу измерения. Чтобы определить межстрочный интервал или высоту строки, браузер попросту умножает это число на размер шрифта. Так, если размер шрифта текста составляет 1 em, а высота строки установлена равной 1,5, то расчетное значение межстрочного интервала равно 1,5 em. В большинстве случаев эффект при указании значения 1,5 em или 150 % идентичен.

Однако, поскольку вложенные элементы наследуют значение свойства `line-height`, часто при использовании процентных значений и единиц em можно столкнуться с проблемами.

Например, вы присваиваете свойству `line-height` элемента `body` значение `150%`. Все элементы веб-страницы унаследуют его. Однако наследуется не процентное, а *рассчитанное* значение межстрочного интервала. Допустим, основной размер шрифта установлен равным 10 пикселам; 150 % от 10 пикселов составляет 15. Все элементы унаследуют межстрочный интервал размером 15 пикселов. Так, если на веб-странице есть абзац текста со шрифтом высотой 36 пикселов, его межстрочный интервал размером 15 пикселов будет намного меньше самого текста, а строки сольются вместе.

В этом примере вместо высоты строки 150 % для элемента `body` лучше установить общий для всех элементов пропорциональный базовый межстрочный интервал в размере 1,5. Любой элемент, вместо того чтобы наследовать точное абсолютное значение высоты строки в пикселах от стиля `body`, умножает размер своего шрифта на этот коэффициент. Так, в вышеупомянутом примере, где абзац текста отображен размером шрифта 36 пикселов, межстрочный интервал составит: 1,5 × 36 = 54 пиксела.

Другими словами, для установки межстрочного интервала вместо процентных значений и единиц em лучше использовать обычные числовые значения.

Выравнивание текста

Одним из самых быстрых способов изменить внешний вид веб-страницы является *выравнивание (выключка)* текста. Используя свойство `text-align`, вы можете расположить абзац в центре веб-страницы, вдоль левого или правого края или выровнять по ширине (формату) (подобно тексту этой книги). Чтобы устанавливать выравнивание для текста, пользуйтесь одним из следующих ключевых слов: `left`, `right`, `justify`, `center`.

```
text-align: center;
```

Выровненный текст замечательно выглядит на печатной странице главным образом потому, что при большой разрешающей способности печати учитываются даже мельчайшие настройки интервалов, а также потому, что большинство программ предпечатной подготовки могут переносить слова (тем самым пытаясь равномерно распределить символы по строкам). Это предотвращает большие промежутки между абзацами. Веб-страницы в форматировании ограничены возможностями намного более грубой регулировки интервалов из-за низкой разрешающей способности мониторов компьютеров и из-за того, что браузеры не поддерживают перенос слов. Когда вы пользуетесь ключевым словом `justify` для выключки по формату, получаются совершенно разные промежутки между словами, текст становится трудночитаемым. Поэтому, применяя такое выравнивание на веб-страницах, убедитесь в том, что получившийся текст читабелен.

Отступ первой строки и изменение интервала между абзацами

В большинстве печатных изданий первая строка каждого абзаца имеет так называемый абзацный отступ. Он выделяет начало каждого абзаца текста, если нет дополнительных промежутков или увеличенных межстрочных интервалов, визуально разделяющих абзацы. В веб-страницах во Всемирной паутине нет отступов, но применяется интервал между абзацами, как в книге.

Если у вас есть желание придать веб-страницам индивидуальность, сделать их непохожими на другие, то воспользуйтесь преимуществами таких свойств CSS, как `text-indent` и `margin`. С их помощью вы можете создать отступ первой строки абзацев и удалить (или увеличить) интервал между ними.

Отступ первой строки

Для установки отступа первой строки абзаца можно использовать такие единицы измерения, как пикселы и em:

```
text-indent: 25px;
```

или

```
text-indent: 5em;
```

Значения в пикселах — абсолютные значения, точное число, в то время как em определяет размер отступа в количестве символов (базируется на текущем размере шрифта).

СОВЕТ

В свойстве абзацного отступа text-indent вы можете использовать отрицательные значения для создания выступа, то есть абзаца с висячей строкой (выступающей влево по отношению к абзацу). Обычно отрицательное значение абзацного отступа используется вместе с указанием значения поля, чтобы отрицательный абзацный отступ не выходил за левую сторону страницы, колонки или блока разметки.

Вы можете также использовать процентные значения, но со свойством `text-indent` эти единицы измерения приобретают другое значение. В данном случае размер выступа, установленный в процентах, связан не со шрифтом текста, а с шириной элемента, в который заключен абзац. Например, если текстовый отступ установлен равным 50 % и абзац охватывает всю ширину окна браузера, то первая строка будет начинаться посередине экрана. Если вы меняете размеры окна, то изменяется ширина абзаца и, соответственно, отступ (про значения свойств, устанавливаемые в процентах, и о том, как они взаимодействуют с шириной элементов веб-страницы, читайте далее в этой главе).

ДЛЯ ОПЫТНЫХ ПОЛЬЗОВАТЕЛЕЙ

Сокращенный метод форматирования текста

Многократный повторный набор свойств стилей — достаточно утомительное занятие, особенно если нужно использовать несколько различных текстовых свойств сразу. На этот случай в языке CSS есть свойство `font`, облегчающее написание стилей. Оно позволяет объединять несколько свойств в одну строку: `font-style`, `font-variant`, `font-weight`, `font-size`, `line-height` и `font-family`. Рассмотрим, к примеру, следующее объявление:

```
font: italic bold small-caps 18px/1.5 Arial, Helvetica, sans-serif;
```

Оно приводит к созданию полужирного курсивного текста, набранного капителью, с размером шрифта 18 пикселов, семейства Arial (или Helvetica, или другого рубленого шрифта), с межстрочным интервалом 150 %. Запомните следующие правила.

- Не обязательно применять все эти свойства, но *нужно* включить размер шрифта и семейство шрифтов:

  ```
  font: 1.5em Georgia, Times, serif;
  ```

- Используйте по одному пробелу после каждого значения свойства, а запятую только для того, чтобы разделить шрифты в списке:

  ```
  Arial, Helvetica, sans-serif
  ```

- Определяя межстрочный интервал, добавляйте слеш после размера шрифта, за которым должно следовать значение межстрочного интервала:

  ```
  1.5em/1.5
  ```

- Последние два свойства должны быть следующими: `font-size` (или `font-size/line-height`), а затем `font-family`; все остальные могут быть перечислены в любом порядке. Например, оба объявления равнозначны и к ним применен одинаковый эффект:

  ```
  font: italic bold small-caps 1.5em Arial;
  font: bold small-caps italic 1.5em Arial;
  ```

- Наконец, исключение значения из списка означает то же, что его установка по умолчанию. Допустим, вы создали стиль `p`, который форматирует все абзацы полужирной курсивной капителью и межстрочным интервалом 2000 % (совсем *не обя-*

зательно это повторять). Затем создали класс с именем, скажем, `.special-Paragraph` с таким объявлением стиля шрифта:

```
font: 1.5em Arial;
```

- После этого применили его к какому-либо абзацу имеющегося текста. В результате теперь наш абзац *не* унаследует полужирную курсивную капитель и межстрочный интервал. Исключение этих четырех значений из стиля `.specialParagraph` можно приравнять к написанию следующего кода:

```
font: normal normal normal 1.5em/normal
Arial;
```

Настройка полей между абзацами

Многие веб-дизайнеры не любят дополнительные поля (так называемый воздух), которые любой браузер добавляет между абзацами. До появления каскадных таблиц стилей с этим приходилось мириться. Теперь можно воспользоваться свойствами `margin-top` и `margin-bottom` для удаления (или увеличения) этих промежутков. Чтобы полностью избавиться от верхнего и нижнего полей, используйте следующий код:

```
margin-top: 0;
margin-bottom: 0;
```

Чтобы удалить поля между *всеми* абзацами веб-страницы, создайте такой стиль:

```
p {
  margin-top: 0;
  margin-bottom: 0;
}
```

Для установки значений абзацных полей, как и для отступов, вы можете применять такие единицы измерения, как пикселы или em. Можно также использовать проценты, но, как и в случае с абзацными отступами, процентные значения относятся к *ширине* элемента, в который заключен абзац. Во избежание путаницы, связанной с вычислением верхнего и нижнего полей, расчет которых базируется на ширине абзацев, проще применять значения в em или пикселах.

ПРИМЕЧАНИЕ

Поскольку не все браузеры обрабатывают верхнее и нижнее поля заголовков и абзацев согласованно, рекомендуется обнулить (то есть удалить) все поля в начале таблицы стилей. Посмотреть на то, как это работает, можно в подразделе «С чистого листа» раздела «Управление каскадностью» главы 5.

Для создания специальных эффектов можно назначить *отрицательное* значение верхнему или нижнему полю между абзацами. Например, верхняя граница, установленная равной 10 пикселам, приподнимает абзац выше на 10 пикселов, возможно даже визуально накладывая его на вышестоящий элемент веб-страницы.

Буквица и форматирование первой строки абзаца

Каскадные таблицы стилей позволяют форматировать абзацы с использованием псевдоэлементов `::first-line` и `::first-letter` (рис. 6.16). С технической точки зрения, это не свойства, а селекторы, определяющие фрагмент, к которому применены

CSS-свойства. С помощью псевдоэлемента `::first-letter` можно отформатировать начальную прописную букву или буквицу, имитируя рукописный стиль текста. Например, чтобы сделать первый символ каждого абзаца полужирным и выделить его красным цветом, необходим следующий код:

```
p::first-letter {
  font-weight: bold;
  color: red;
}
```

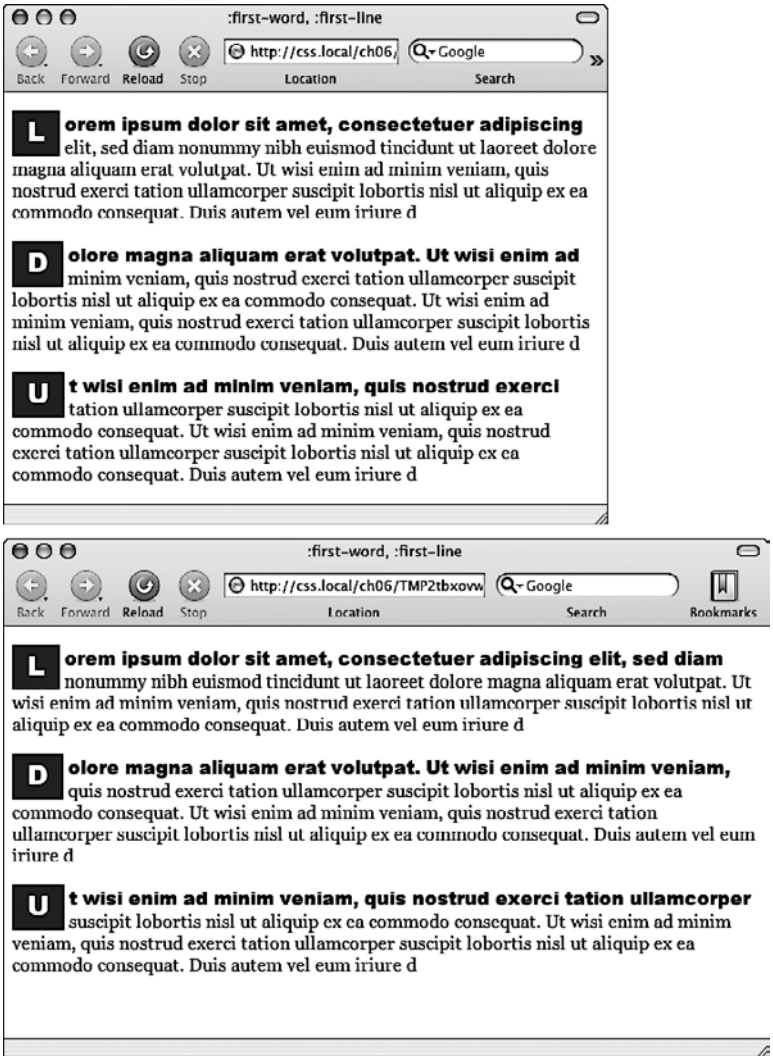

Рис. 6.16. Псевдоэлемент ::first-letter изменяет первый символ форматируемого элемента, например заглавные буквы абзацев. С другой стороны, селектор ::first-line определяет стиль первой строки абзаца. Даже если посетители вашего сайта изменят размер окна *(внизу)*, браузер будет форматировать каждое слово в первой строке абзаца

Для избирательного форматирования, скажем, только первого символа определенного абзаца можно применить класс, например .intro:

```
<p class="intro">Текст вводного абзаца начинается здесь...</p>
```

Затем вы можете создать стиль .intro::first-letter. Псевдоэлемент ::first-line форматирует первую строку абзаца. Вы можете применить его к любому фрагменту текста, будь то заголовок (h2::first-line) или абзац (p::first-line). Как и в случае с ::first-letter, можно применить класс к единственному абзацу и отформатировать только его первую строку. Допустим, вы хотите отобразить прописными буквами первую строку первого абзаца на странице. Примените класс к HTML-коду первого абзаца: <p class="intro"> — и напишите следующий код:

```
.intro::first-line { text-transform: uppercase; }
```

ПРИМЕЧАНИЕ

По какой-то странной причине браузеры Chrome и Safari не распознают свойство text-transform, когда оно используется с псевдоэлементом ::first-line. Другими словами, вы не можете применять в Chrome и Safari каскадные таблицы стилей для преобразования букв первой строки абзаца в прописные.

Форматирование списков

Элементы ul и ol создают маркированные и нумерованные списки: взаимосвязанных элементов, пунктов или пронумерованных шагов, последовательности действий. Однако, как вы, наверное, заметили, не всегда подходит предопределенный браузером способ форматирования. Возможно, вы захотите заменить в маркированных списках стандартный маркер собственным, более красивым, использовать в нумерованных списках буквы вместо чисел и т. д.

Типы списков

Большинство браузеров отображают маркированные списки (элементы ul), используя маркеры в виде окружности, а нумерованные списки (ol) — предваряя пункты числами. С помощью каскадных таблиц стилей вы можете выбрать маркеры трех типов: disc (сплошной кружок), circle (полый кружок), square (сплошной квадрат). Для нумерованных списков предусмотрено шесть вариантов-схем нумерации: decimal, decimal-leading-zero, upper-alpha, lower-alpha, upper-roman, lower-roman (рис. 6.17). Все эти варианты можно выбрать, используя свойство list-style-type каскадных таблиц стилей:

```
list-style-type: square;
```

или

```
list-style-type: upper-alpha;
```

Числа можно заменить буквами греческого алфавита — α, β, γ, воспользовавшись значением lower-greek. Существует множество других схем нумерации, включая армянский, грузинский, катакана и другие региональные варианты. Информацию о них можно найти по адресу tinyurl.com/pmkpsj9.

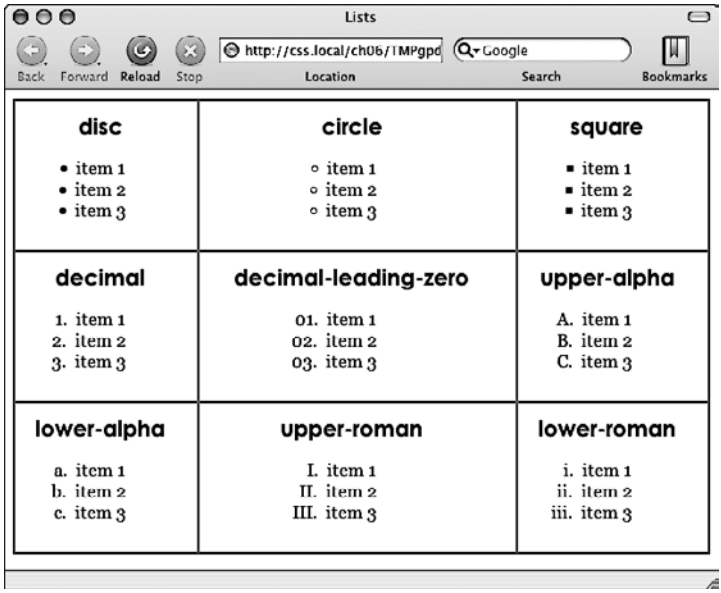

Рис. 6.17. Каскадные таблицы стилей предоставляют множество различных способов маркировки нумерованных и ненумерованных списков, начиная с набора геометрических фигур и заканчивая различными системами счисления

Чаще всего это свойство используют при определении стилей, форматирующих элементы `ol` или `ul`. Типичные примеры — включение свойства в стили тега `ol` или `ul`: `ul { list-style-type: square; }` либо в класс, применяемый к одному из этих элементов. Тем не менее вы можете применить это свойство и к отдельно взятому элементу списка (`li`). Вы даже можете применить несколько стилей с различными маркерами к отдельным пунктам-элементам одного и того же списка. Например, можете создать стиль тега `ul`, устанавливающий маркеры в виде квадратов, а затем класс `.circle`, который изменяет тип маркера на полые кружки:

```
li {list-style-type: square; }
.circle { list-style-type: circle; }
```

Теперь примените класс к элементам списка через один для чередования квадратных и круглых маркеров:

```
<ul>
<li>Элемент 1</li>
<li class="circle">Элемент 2</li>
<li>Элемент 3</li>
<li class="circle">Элемент 4</li>
</ul>
```

Или, используя рассмотренный в главе 3 селектор `nth-of-type`, можно вообще избавиться от имени класса:

```
li {list-style-type: square; }
li:nth-of-type(odd) { list-style-type: circle; }
```

Иногда может понадобиться скрыть маркеры, например, если вы захотите использовать собственные графические значки (см. практикум этой главы). Кроме того, когда панель навигации сайта представляет собой список ссылок, вы также можете использовать список `ul`, скрыв его маркеры (см. подраздел «Использование маркированных списков» раздела «Создание панелей навигации» главы 9). Чтобы отключить отображение маркеров, используйте ключевое слово `none`:

```
list-style-type: none;
```

Позиционирование маркеров и нумерации списков

Как правило, браузеры отображают маркеры или числа слева от текста элементов списка (рис. 6.18, *слева*). Благодаря каскадным таблицам стилей вы можете управлять размещением маркеров, используя свойство `list-style-position`. Установить местоположение можно вне (стандартный способ отображения браузерами, см. рис. 6.18, *слева*) или внутри текстовых блоков элементов списка (см. рис. 6.18, *справа*):

```
list-style-position: outside;
```

или

```
list-style-position: inside;
```

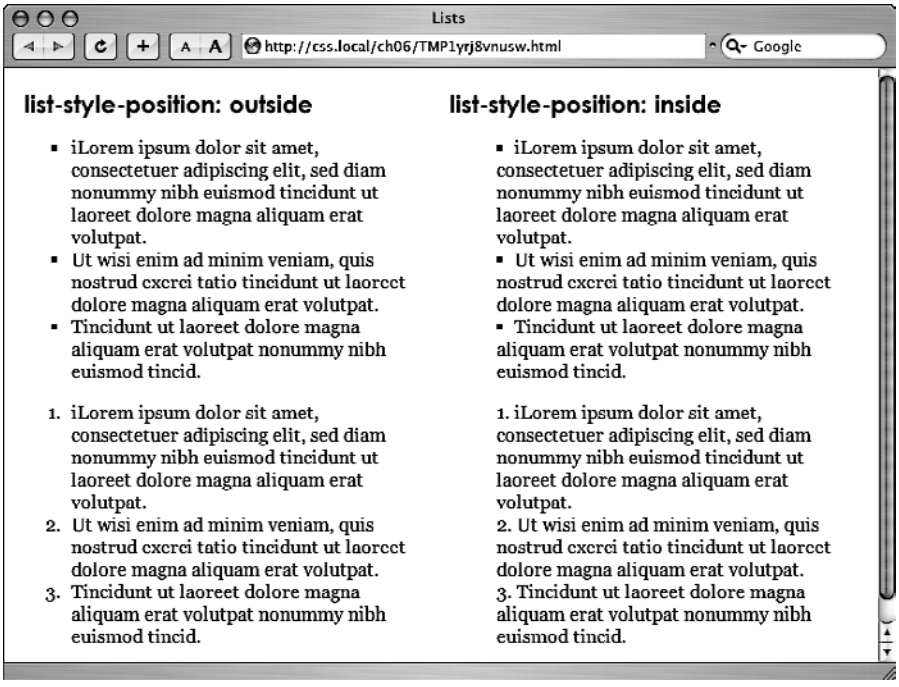

Рис. 6.18. Используя свойство list-style-position, вы можете управлять позиционированием маркеров и чисел в списках. Значение outside визуально выделяет список и каждый его элемент из всего текста. Значение inside обеспечит списку максимальную ширину на странице

СОВЕТ

Вы можете изменить промежуток между маркером и текстом, увеличив или уменьшив значение свойства padding-left (см. раздел «Управление размерами полей и отступов» главы 7). Чтобы использовать его, вам нужно создать стиль для элементов li. Этот способ работает только в том случае, если свойство list-style-position определено со значением outside (или вообще отсутствует).

Кроме того, если вам не нравится, что браузеры создают для списков отступ, смещая все содержимое вправо, можете переопределить стиль, присвоив свойствам `margin-left` и `padding-left` значение `0`. Чтобы удалить отступ, можно создать следующий групповой селектор:

```
ul, ol {
  padding-left: 0;
  margin-left: 0;
}
```

Вы можете создать класс с такими свойствами и применить его к конкретным элементам `ul` или `ol`. Рекомендуется указать значения обоих свойств, `padding` и `margin`, по той причине, что одни браузеры для управления отступом используют свойство `padding` (Firefox, Mozilla, Safari), а другие — `margin` (Internet Explorer). Подробно про свойства `padding` и `margin` вы можете прочесть в следующей главе.

В обычном порядке браузеры отображают пункты маркированных списков друг за другом, без дополнительного промежутка. Добавить интервал между ними можно, применяя свойства `margin-top` и `margin-bottom` к конкретным элементам списка. Они работают с интервалом между пунктами списков точно так же, как с абзацами. Единственное, вы должны удостовериться в том, что стиль применяется к элементу `li`: создайте класс и примените его индивидуально к каждому элементу. Стиль *не* должен применяться к элементам `ul` или `ol`. Добавление верхнего или нижнего полей к этим элементам увеличивает промежуток между всем списком и абзацами выше или ниже его. На интервал между элементами они не оказывают никакого влияния.

Графические маркеры

Если вам недостаточно стандартных маркеров квадратной и круглой формы, вы можете создать свои собственные. Используя графический редактор, например Photoshop или Fireworks, можно быстро создать красочные и интересные маркеры. В качестве источников также можно рассмотреть встроенные в программы клипарты и символьные шрифты (такие как Webdings).

Свойство `list-style-image` позволяет определить путь к графическому символу на сервере таким же образом, как вы указываете местонахождение файла с изображением, используя атрибут `src` HTML-элемента `img`. Синтаксис команды следующий:

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

Значение `url` и круглые скобки обязательны. Часть, заключенная в круглые скобки, — в данном примере `images/bullet.gif` — это и есть путь к графическому символу. Обратите также внимание на то, что путь в кавычки заключать не нужно.

ПРИМЕЧАНИЕ

Указывая путь или адрес к графическим файлам (изображениям, графическим символам) во внешней таблице стилей, имейте в виду, что путь должен указываться относительно таблицы стилей, а не веб-страницы. Более подробно об этом вы прочтете в разделе «Добавление фоновых изображений» главы 8, когда мы начнем использовать графику.

Свойство `list-style-image` позволяет использовать графические символы в качестве маркеров. Однако оно не позволяет управлять их положением. Маркер может оказаться слишком высоко или низко расположенным относительно пункта списка. Придется редактировать сам графический символ маркера, пока он не будет сочетаться со списком. О более грамотном подходе вы узнаете в главе 8. Он основан на использовании свойства `background-image`. Это свойство позволяет точно позиционировать графические элементы, в том числе маркеры в списках.

ПРИМЕЧАНИЕ

Как и в случае со свойством font (см. врезку «Для опытных пользователей» ранее в этой главе), может применяться сокращенный метод указания атрибутов списков. В свойстве list-style могут быть перечислены все настройки форматирования списка, в том числе list-style-image, list-style-position и list-style-type. Например, стиль ul { list-style: circle inside; } отобразит списки с маркерами в виде полых кружков, расположив их внутри списка. Когда вы описываете стиль списка с одновременным указанием типа маркера и графического символа — list-style: circle url(images/bullet.gif) inside; — и графический символ не может быть обнаружен по заданному пути, браузер будет использовать предопределенный маркер, в данном случае полый кружок.

ЧАВО

Настройка маркеров и чисел в списках

Я хочу отформатировать числа нумерованных списков так, чтобы они отображались полужирным шрифтом красного цвета вместо надоевшего черного. Как можно настроить внешний вид маркеров и чисел?

Каскадные таблицы стилей предлагают несколько способов настройки параметров маркеров, предваряющих пункты — элементы списка. Вы можете использовать собственные графические символы, как описано выше. Более совершенный способ, экономящий объем CSS-кода, состоит в том, чтобы использовать так называемый *сгенерированный контент*. Этим термином обозначаются, по сути, элементы, отсутствующие в исходном коде и автоматически добавляемые браузером при отображении страницы. Хороший пример — сами маркеры. Вы не указываете значки маркеров при создании списка — они добавляются на веб-страницу автоматически. С помощью каскадных таблиц стилей можно сообщить браузеру, чтобы он генерировал, добавлял такое содержимое и даже форматировал должным образом все, что находится перед текстом пунктов списка — `li`. Вы узнали о генерируемом контенте из главы 3 и теперь можете сделать обычные маркеры списка красными, добавив в свою таблицу стилей следующий CSS-код:

```
ul li {
  list-style-type: none;
}
ul li:before {
  content: counter(item, disc) " ";
  color: red;
}
```

А если нужно красным цветом отформатировать числа нумерованного списка, можно добавить такой CSS-код:

```
ol li {
  list-style-type: none;
  counter-increment: item;
}
ol li:before {
  content: counter(item) ". ";
  color: red;
}
```

Как придать стиль нумерованному списку, подробно написано по адресу tinyurl.com/qcevfy7.

Практикум: форматирование текста

В этом разделе мы попрактикуемся в форматировании таких элементов веб-страниц, как заголовки, списки, абзацы текста, используя мощные средства языка CSS.

Чтобы начать обучение, вы должны иметь в распоряжении файлы с учебным материалом. Для этого нужно загрузить файлы для выполнения заданий практикума, расположенные по адресу github.com/mrightman/css_4e. Перейдите по ссылке и загрузите ZIP-архив с файлами (нажав кнопку Download ZIP в правом нижнем углу страницы). Файлы текущего практикума находятся в папке 06.

Настройка параметров страницы

Начнем с таблицы стилей и добавим правило `@font-face`, чтобы загрузить некоторые веб-шрифты для форматирования основного текста страницы.

1. Запустите браузер и откройте файл `text.html` (рис. 6.19).

   Здесь пока особо не на что смотреть — есть только несколько заголовков, абзацев и один маркированный список. Но скоро вы заметно преобразите эту страницу.

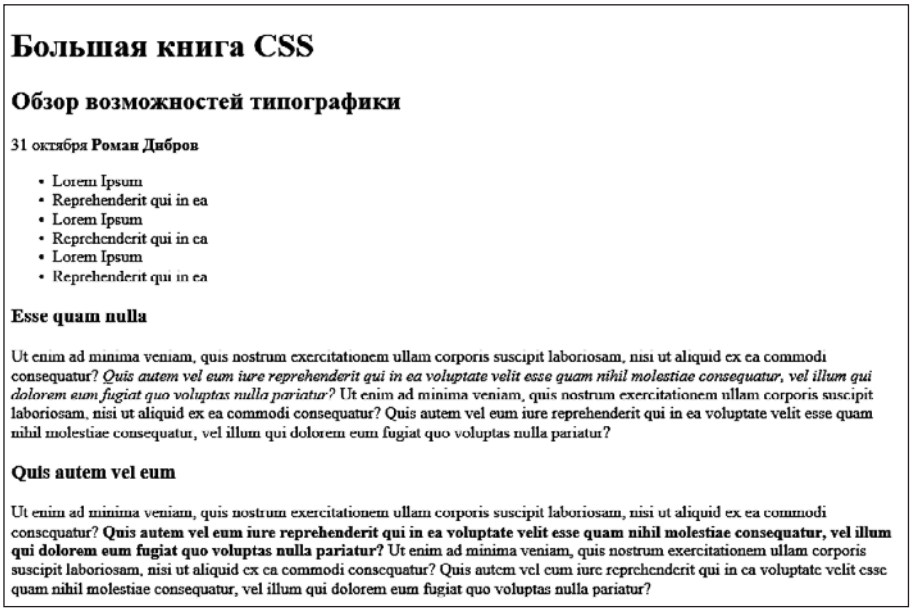

Рис. 6.19. Работа над любой веб-страницей начинается с обычного HTML-контента

2. Откройте файл `text.html` в редакторе HTML-кода.

   В этом проекте вы будете использовать как собственные файлы шрифтов, так и службу Google Fonts. Начнем с добавления ссылки на шрифт Google.

3. В разделе заголовка веб-страницы щелкните кнопкой мыши сразу после закрывающего тега `</title>`. Нажмите клавишу Enter и наберите код:

```
<link href="http://fonts.googleapis.com/css?family=Slabo+27px" rel="stylesheet">
```

Этот код говорит серверу Google отправить вам антиквенный (с засечками) шрифт Slabo размером 27px. Далее вы добавите ссылку на внешнюю таблицу стилей, которая будет основой для стилей данного урока.

4. После только что добавленного элемента link введите следующий код:

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

Вы только что связали HTML-файл с внешней таблицей стилей, расположенной в папке css. В этой папке находятся внешняя таблица стилей и веб-шрифты.

5. Откройте файл styles.css, расположенный в папке css. Эта таблица стилей содержит базовый набор сброшенных стилей (см. раздел «Управление каскадностью» главы 5).

Если вы просмотрите файл text.html в браузере, то увидите, что текст на странице (заголовки, абзацы и т. д.) выглядит примерно одинаково, то есть все стандартное HTML-форматирование браузера было устранено и теперь можно начинать с чистого листа.

Затем нужно добавить необходимые правила @font-face для загрузки четырех веб-шрифтов. Все они описывают один и тот же шрифт PTSans, но включают полужирное, курсивное и полужирно-курсивное начертание.

6. В верхней части файла styles.css добавьте следующий код (перед кодом сброса стилей) (если не хотите вводить весь этот код, откройте файл at-font-face.css, расположенный в папке с учебными файлами, скопируйте и вставьте его в файл styles.css):

```
@font-face {
  font-family: 'PTSans';
  src: url('fonts/PTSansRegular.woff2') format('woff2'),
  url('fonts/PTSansRegular.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}
@font-face {
  font-family: 'PTSans';
  src: url('fonts/PTSansItalic.woff2') format('woff2'),
  url('fonts/PTSansItalic.woff') format('woff');
  font-weight: normal;
  font-style: italic;
}
@font-face {
  font-family: 'PTSans';
  src: url('fonts/PTSansBold.woff2') format('woff2'),
  url('fonts/PTSansBold.woff') format('woff');
  font-weight: bold;
  font-style: normal;
}
@font-face {
  font-family: 'PTSans';
  src: url('fonts/PTSansBoldItalic.woff2') format('woff2'),
  url('fonts/PTSansBoldItalic.woff') format('woff');
```

```
  font-weight: bold;
  font-style: italic;
}
```

Если вкратце, то вы создали новое семейство шрифтов PTSans, которое сможете использовать в любых создаваемых вами новых стилях. Для добавления полужирно-курсивных вариантов используется метод, рассмотренный ранее. Суть его в том, что в области применения обычного элемента полужирного начертания (например, в заголовках или при использовании элемента `strong`) браузер будет автоматически переключаться на полужирную версию шрифта.

Вы добавили правило `@font-face` в начале внешней таблицы стилей. Не обязательно, но рекомендуется поместить все шрифты в верхней части, а стили — в нижней.

Затем вы создадите стиль, определяющий ряд общих свойств для всего текста на странице.

7. В нижней части таблицы стилей, после стилей сброса, перейдите на новую строку и введите `html {`.

   Это основной селектор тега, применяемый к элементу `html`. В главе 4 мы выяснили, что прочие элементы наследуют свойства этого элемента. Можно настроить ряд основных характеристик текста, таких как шрифт, цвет и размер для их использования последующими элементами в качестве их исходных настроек.

8. Снова нажмите клавишу Enter и добавьте следующие два свойства:

   ```
   font-family: PTSans, Arial, sans-serif;
   font-size: 62.5%;
   ```

   Эти две инструкции устанавливают шрифт PTSans (или Arial, если браузер не может загрузить PTSans) и значение для размера шрифта `62.5%`.

ПРИМЕЧАНИЕ

Зачем нужно устанавливать базовый размер шрифта веб-страницы равным 62,5 %? Оказывается, если умножить 62,5 % на 16 пикселов (стандартный размер шрифта текста большинства браузеров), получится 10 пикселов. При таком начальном размере шрифта очень просто вычислить размеры всех последующих и представить, как они будут выглядеть на экране монитора. Например, 1,5em будет равняться 1,5 × 10, или 15 пикселов, 2em — 20 пикселов и т. д. Считать очень просто, умножая размер в em на 10. Вы также можете использовать единицы rem (описанные в разделе «Форматирование символов и слов» данной главы), которые похожи на em, но позволяют избежать проблемы, связанной с наследованием вложенных единиц em.

9. Завершите определение текущего стиля, нажав клавишу Enter и набрав закрывающую фигурную скобку для указания окончания стиля.

   На данном этапе стиль должен выглядеть следующим образом:

   ```
   html {
     font-family: PTSans, Arial, sans-serif;
     font-size: 62.5%;
   }
   ```

   Ваша таблица стилей закончена.

10. Сохраните файл styles.css и откройте файл text.html для просмотра в браузере, чтобы увидеть результат работы.

    Текст на странице изменил свой цвет и шрифт. Он стал действительно маленьким. Не волнуйтесь, это из-за размера 62,5 %, который вы установили на шаге 8. Это начальный параметр для всего текста, и вы сможете спокойно увеличить текст, определяя размеры в единицах em для других элементов.

Форматирование заголовков и абзацев

Теперь, когда основное форматирование текста выполнено, пришло время улучшить вид заголовков и абзацев.

1. Вернитесь к файлу styles.css в редакторе HTML-кода. Установите курсор после закрывающей скобки селектора тега html, добавьте новую строку и наберите код .main h1 {.

    Это *селектор потомков*, более специфичный по сравнению с базовым селектором тега html. В данном случае селектор указывает браузеру «применить соответствующее форматирование» к любому элементу h1, находящемуся внутри другого элемента с классом main. Если вы просмотрите HTML-код веб-страницы, то увидите, что там присутствует элемент div с классом main ( ). Как вы узнаете позже, при разметке дизайнов, основанной на каскадных таблицах стилей, достаточно распространено группирование HTML-элементов внутри контейнеров div. Вы сможете размещать отдельные элементы div для создания колонок и других блоков разметки страниц. Распространено также использование селекторов потомков наподобие этого для точного определения свойств форматирования с воздействием лишь на элементы в определенных областях страницы.

2. Нажмите клавишу Enter и наберите три свойства:

```
color: rgb(249,212,120);
font-family: "Arial Black", Arial, Helvetica, sans-serif;
font-size: 4em;
```

    Вы только что изменили цвет заголовков h1, а также их шрифт. На этот раз вместо веб-шрифта указан шрифт, который может быть установлен на компьютере посетителя. Шрифт Arial Black установлен на многих компьютерах, но если у какого-нибудь посетителя он отсутствует, браузер воспользуется шрифтом Arial или Helvetica либо другим рубленым шрифтом. Кроме того, вы установили размер шрифта равным 4em, что для большинства браузеров составляет 40 пикселов (если, конечно, посетитель не изменял параметры шрифтов в настройках своего браузера). Это все благодаря значению 62.5%, установленному ранее на шаге 7. Таким образом, базовый размер шрифта стал составлять 10 пикселов в высоту, а вычисление 4 × 10 задает размер 40 пикселов. Затем к заголовку будет добавлена тень.

3. Завершите текущий стиль, нажав клавишу Enter и добавив код, выделенный ниже полужирным шрифтом (не забудьте указать закрывающую скобку):

```
.main h1 {
  color: rgb(249,212,120);
  font-family: "Arial Black", Arial, Helvetica, sans-serif;
```

```
  font-size: 4em;
  text-shadow: 4px 4px 6px rgba(0,0,0,.75);
}
```

Выделенной строкой кода добавлена тень, смещенная на 4 пиксела вправо, на 3 пиксела ниже и с размытостью 6 пикселов. Кроме этого, используется RGBA-цвет, устанавливающей для тени черный цвет и 75%-ный уровень прозрачности.

4. Сохраните файл и просмотрите HTML-страницу в браузере.

   Теперь изменим внешний вид остальных заголовков и абзацев.

ПРИМЕЧАНИЕ

При открытии файла браузер сохраняет его в хранилище на жестком диске компьютера посетителя, которое называется кэшем. Если браузеру вновь потребуется тот же файл, то вместо повторной загрузки с сервера он обратится к нему в кэше. Однако кэшированный файл может быть неактуален. Браузер может загрузить старую кэшированную версию вашей внешней таблицы стилей вместо обновленной. Если после обновления внешней таблицы стилей страница выглядит по-прежнему, в браузере нажмите клавишу F5 или сочетание клавиш Ctrl+Shift+R (⌘+Shift+R), чтобы загрузить обновленную версию файла.

5. Вернитесь к файлу `styles.css` в редакторе HTML-кода. Установите курсор после закрывающей скобки стиля `.main h1`, нажмите клавишу Enter и добавьте следующие стили:

```
.main h2 {
  font: normal 3.5em "Slabo 27px", Garamond, Times, serif;
  color: rgb(37,76,143);
  border-bottom: 1px solid rgb(200,200,200);
  margin-top: 25px;
}
```

Здесь используется еще один селектор потомков, который относится только к заголовкам `h2`, находящимся внутри другого элемента с классом `main`. Свойство `font` сокращает количество кода, объединяя в себе правила `font-weight`, `font-size` и `font-family`. Другими словами, одна строка кода форматирует заголовок обычным начертанием, устанавливает для него высоту `3,5em` и определяет шрифт.

Кроме того, этот стиль изменяет цвет заголовков, добавляет декоративную границу под заголовком и немного пространства над заголовком.

Пришло время взяться за другие заголовки.

6. Добавьте новый стиль под тем, который вы создали на предыдущем шаге:

```
.main h3 {
  color: rgb(241,47,6);
  font-size: 1.9em;
  font-weight: bold;
  text-transform: uppercase;
  margin-top: 25px;
  margin-bottom: 10px;
}
```

Он настраивает некоторые свойства обычного форматирования: цвет, размер шрифта, насыщенность, а также использует свойство `text-transform`, чтобы текст заголовка `h3` был оформлен прописными буквами. Наконец, он добавляет немного пространства сверху и снизу заголовка с помощью свойства `margin`.

Далее вы улучшите внешний вид абзацев.

7. Добавьте новый стиль на страницу:

```
.main p {
  font-size: 1.8em;
  line-height: 1.5;
  margin-left: 150px;
  margin-right: 50px;
  margin-bottom: 10px;
}
```

Этот стиль содержит свойство `line-height`, которое устанавливает расстояние между строками. Значение `1.5` является коэффициентом: для вычисления высоты строки размер шрифта (`1.75em`) умножается на величину 1,5. Обычно коэффициент увеличивает высоту строки в 1,5 раза, или на 150 % относительно размера шрифта. Стиль увеличивает интервал между строками абзаца. Благодаря этому дополнительному пространству текст располагается свободнее, а предложения становится легче читать (но только если вы используете латиницу).

Этот стиль также увеличивает размер шрифта до 1,8 em (18 пикселов для большинства браузеров) и смещает абзац от левого и правого краев страницы. Вы можете заметить, что для свойства `margin` приходится набирать слишком много кода. На этот случай, как вы узнаете в следующей главе, есть сокращенная запись свойства.

Теперь попробуем более совершенный тип селектора.

8. Добавьте следующий стиль в таблицу стилей:

```
.main p::first-line {
  font-weight: bold;
  color: rgb(153,153,153);
}
```

Псевдоэлемент `::first-line` воздействует только на первую строку абзаца. В этом случае именно первая строка текста в каждом абзаце внутри контейнера `div` с классом `main` будет окрашена в серый цвет и выделена полужирным.

9. Сохраните файл `styles.css` и откройте файл `text.html` для просмотра результата в браузере.

На текущий момент ваша веб-страница должна быть похожа на показанную на рис. 6.20.

Форматирование списков

На этой странице есть один маркированный список. Вы переместите его вверх к правому краю страницы и сделаете так, чтобы текст, расположенный после

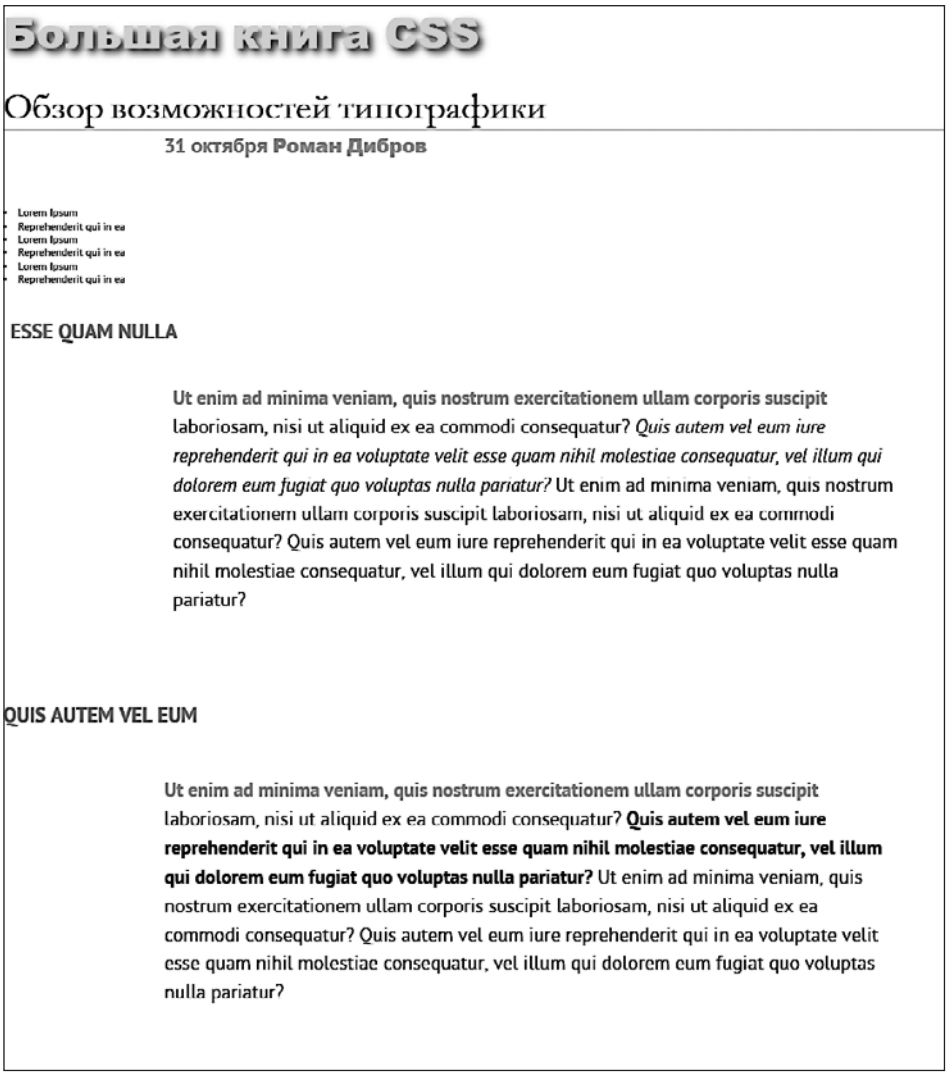

Рис. 6.20. Внешний вид веб-страницы преображается: заголовки, абзацы и основной текст приведены в порядок. В зависимости от шрифтов, установленных на вашем компьютере, результат, получившийся у вас, может несколько отличаться

этого списка, обтекал его. С помощью каскадных таблиц стилей сделать это достаточно легко.

1. Вернитесь к файлу `styles.css` в редакторе HTML-кода. Добавьте следующий стиль в конце таблицы стилей:

```
.main ul {
  margin: 50px 0 25px 50px;
  width: 25%;
```

```
  float: right;
}
```

При форматировании списков обычно создаются стили для двух разных элементов: непосредственно для самого списка (элемент `ul` для маркированных либо `ol` для нумерованных списков) и отдельных элементов списка (элемент `li`). Этот стиль управляет всем списком.

ПРИМЕЧАНИЕ

В Firefox страница может отображаться в искаженном виде. В этом случае попробуйте использовать другой браузер, например актуальную версию Internet Explorer.

В стиле выполняется несколько действий. Во-первых, свойство `margin` используется в сокращенной записи. Одна строка кода устанавливает все четыре поля вокруг списка, замещая четыре отдельных свойства полей (`margin-top`, `margin-right` и т. д.). Четыре значения представлены в следующем порядке: верхняя сторона, правая, нижняя и левая. Таким образом, этот стиль устанавливает поле шириной 50 пикселов сверху списка, 0 — справа, 25 — снизу и 50 пикселов — слева.

Свойство `width` задает для всего списка ширину, равную 25 % от ширины окна браузера. Если какой-нибудь пункт списка содержит больше текста, чем может поместиться в пределах этого пространства, он переходит к другой строке.

Свойство `float` по-настоящему волшебное: в данном случае `float: right` означает перемещение списка к правому краю страницы. Это свойство также приводит к тому, что следующий за списком текст обтекает список слева. Это замечательный метод, более подробно о плавающих элементах вы узнаете в главе 7.

Далее вы определите внешний вид отдельных пунктов списка.

2. Добавьте еще один стиль в таблицу:

```
.main li {
  color: rgb(32,126,191);
  font-size: 1.5em;
  margin-bottom: 7px;
}
```

Здесь нет ничего нового: обычное изменение цвета и размера, а также добавление пространства под каждым пунктом списка. Пришло время взглянуть на результат.

ПРИМЕЧАНИЕ

Если вы хотите добавить пространство между пунктами списка, необходимо добавить верхние или нижние поля для элемента li. Добавление полей к элементам ul или ol увеличит пространство вокруг всего списка.

3. Сохраните страницу и воспользуйтесь предварительным просмотром в браузере.

Теперь страница должна быть похожа на ту, что представлена на рис. 6.21.

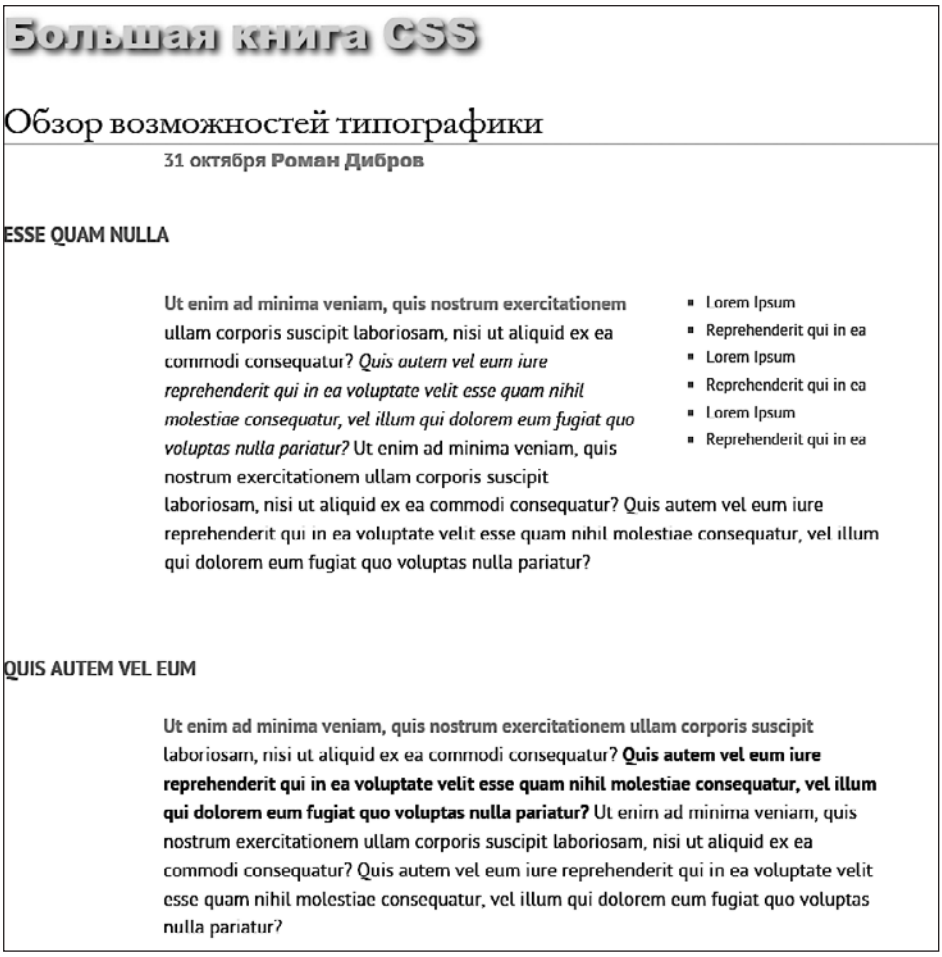

Рис. 6.21. Свойство float дает интересные возможности при проектировании дизайна. В данном случае маркированный список перемещается к правому краю страницы. Фактически свойство float настолько полезно, что является основным при компоновке макетов на базе CSS, пример чего мы рассмотрим в главе 12

Точная настройка с классами

Иногда появляется желание иметь еще больше контроля над тем, как применяется стиль. Например, вы хотите видеть большинство абзацев в каком-либо разделе страницы одинаковыми, но, вероятно, пожелаете определить уникальный вид для одного или двух из них. В этом примере абзац рядом с верхней частью страницы — November 30 Rod Dibble — содержит особую информацию (о дате публикации и об авторе). Сделаем так, чтобы этот абзац выделялся среди других, добавив класс в HTML-код и создав для него стиль.

1. Откройте файл `text.html` в редакторе HTML-кода. Найдите вышеупомянутый абзац в HTML-коде (`<p>31 октября <strong>Роман Дибров</strong></p>`)

и добавьте `class="byline"` к открывающему тегу `<p>`. HTML-код должен выглядеть так:

```
<p class="byline">31 октября <strong>Роман Дибров</strong></p>
```

Теперь осталось создать класс, замещающий общие свойства форматирования абзацев на этой странице.

2. Откройте файл `styles.css` в редакторе HTML-кода. В нижней части таблицы стилей добавьте следующий код:

```
.main .byline {
  font-size: 1.6em;
  margin: 5px 0 25px 50px;
}
```

Стиль настраивает размер и расположение только одного абзаца.

Обратите внимание, что, если бы вы назвали этот стиль `.byline` (простой селектор класса), он бы не работал. Благодаря правилам каскадности, описанным в предыдущей главе, `.byline` менее специфичен, чем стиль `.main p`, созданный в шаге 7 ранее, поэтому он будет неспособен заместить размер и поля, указанные в стиле `.main p`. А стиль `.main .byline`, в свою очередь, более специфичен.

Форматирование этого абзаца по-прежнему требует доработки. Было бы замечательно, если бы имя выделялось еще больше. В этом случае нам поможет HTML-разметка.

3. Добавьте следующий код в таблицу стилей:

```
.main .byline strong {
  color: rgb(32,126,191);
  text-transform: uppercase;
  margin-left: 11px;
}
```

Если вы просмотрите HTML-код в шаге 1, то увидите, что имя окружено тегами элемента `strong`. Он используется для того, чтобы выделить текст и пометить его как важный. Но это не значит, что вам нужно позволять ему быть полужирным, как большинство браузеров отображают этот элемент по умолчанию. Напротив, селектор потомков относится к элементу `strong`, но только в тех случаях, когда он появляется внутри другого элемента с классом `.byline` и лишь если они все находятся внутри еще одного элемента с классом `main`, — вот так, очень специфично.

Этот стиль форматирует шрифт синим цветом, превращает буквы в прописные и добавляет немного пространства с левой стороны (немного отодвигая имя от текста November 30).

Последние штрихи

Заключительная шлифовка дизайна веб-страницы будет заключаться в добавлении нескольких стилей, форматирующих страницу и контейнер `div` с классом `main`, чтобы они выглядели лучше.

Кроме того, здесь вы реализуете несколько интересных типографических приемов.

1. Вернитесь к файлу styles.css в редакторе HTML-кода.

   Сначала зададим фоновый цвет и изображение для страницы.

2. Найдите стиль html в верхней части таблицы стилей и добавьте новое свойство (изменения выделены полужирным шрифтом):

```
html {
  font-family: PTSans, Arial, sans-serif;
  font-size: 62.5%;
  background: rgb(225,238,253) url(../images/bg_body.png) repeat-x;
}
```

   Свойство background представляет собой мощный инструмент для любого веб-дизайнера. Вы уже использовали его пару раз в предыдущих практикумах; оно позволяет добавлять цвет, а также изображения и управлять их расположением в области какого-либо HTML-элемента.

   Вы узнаете все тонкости этого свойства в следующей главе, а сейчас учтите, что добавленная строка кода изменяет цвет фона страницы на светло-голубой и добавляет темно-синюю полосу в верхнюю часть страницы.

   Далее мы преобразим элемент div с классом main.

3. Добавьте еще один стиль между html и .main h1:

```
.main {
  max-width: 740px;
  margin: 0 auto;
  padding: 0 10px;
  border: 4px solid white;
  background: transparent url(../images/bg_banner.jpg) no-repeat;
}
```

   Щелкните кнопкой мыши после закрывающей скобки } стиля html, нажмите клавишу Enter и введите код, представленный выше. Вам не обязательно создавать стиль именно в этом месте, чтобы он исправно работал. Однако кажется логичным размещение стиля, управляющего элементом div, перед другими стилями, которые форматируют элементы внутри контейнера div.

   Свойство max-width устанавливает максимальную ширину данного контейнера div. Это значит, что если у посетителя страницы ширина экрана меньше 740 пикселов, то элемент div сожмется. Однако его ширина никогда не превысит 740 пикселов. Значения свойства margin здесь — 0 auto — добавляют 0 пикселов пространства над и под контейнером div и устанавливают для правого и левого полей параметр auto, располагающий этот элемент в центре окна браузера.

   Свойство padding добавляет пространство внутри контейнера, смещая контент внутри элемента div от его границ. Наконец, мы также поместили изображение в качестве фона контейнера div.

Два последних стиля не имеют ничего общего с форматированием текста, но, если вы просмотрите страницу, то увидите, что благодаря им он выглядит намного лучше, за исключением двух заголовков. Первый из них недостаточно жирный, а второй должен появляться под недавно добавленным рисунком.

4. Добавьте следующий код после стиля `.main h1`:

```
.main h1 strong {
  font-size: 150px;
  color: white;
  line-height: 1;
  margin-right: -.5em;
}
```

HTML-код заголовка выглядит следующим образом:

```
<h1><strong>CSS</strong> The Missing Manual</h1>
```

Аббревиатура CSS заключена в теги элемента `strong`, поэтому данный селектор потомков форматирует только этот текст (здесь он подобен стилю, который вы добавили в шаге 3 ранее). Размер шрифта был увеличен, его цвет изменился, а высота строки задана так, что текст теперь вписывается в верхнюю часть страницы.

Вы заметите, что высота строки равна `1`, а, как вы читали в начале этой книги, единица em основывается на текущем размере шрифта элемента, поэтому в данном случае высота строки будет составлять 150 пикселов — таков размер шрифта данного стиля.

Еще один интересный прием позволяет осуществить свойство `margin-right`, которому присвоено отрицательное значение `-.5em`. Поскольку положительные значения размеров полей отодвигают элементы друг от друга, отрицательные, в свою очередь, сближают. В данном случае остальной текст заголовка (The Missing Manual) располагается рядом с аббревиатурой CSS.

ПРИМЕЧАНИЕ

Использование отрицательных значений размеров полей разрешено в CSS (хотя это и неочевидный прием).

5. Сохраните таблицу `styles.css` и просмотрите файл `text.html` в браузере.

Веб-страница должна иметь вид, показанный на рис. 6.22. Теперь вы можете сравнить получившийся файл с законченным вариантом `text.html`, который находится в папке 06_finished учебного материала.

Вы изучили основные способы форматирования текста, предлагаемые каскадными таблицами стилей, и превратили малопривлекательный текст в страницу с отличным дизайном.

В следующей главе мы рассмотрим размещение на веб-страницах графики, границ, полей и применение прочих мощных команд форматирования, предлагаемых каскадными таблицами стилей.

Рис. 6.22. С небольшой помощью языка CSS можно превратить простой текст в документ с профессиональным дизайном