

используется система модульной верстки Skeleton, но вы можете выбрать любую другую по своему усмотрению (см. врезку далее).

ИНФОРМАЦИЯ ДЛЯ ОПЫТНЫХ ПОЛЬЗОВАТЕЛЕЙ

Основные системы модульной верстки

В этой главе используется простая, но мощная система модульной верстки под названием Skeleton (getskeleton.com). Но она не одна в своем роде.

- Simple Grid (tinyurl.com/d3vwkuu) — еще одна простая система модульной верстки. Это *каркас* — она не обеспечивает создание никакого CSS-кода, кроме модульной сетки. Эта система адаптивная, то есть дизайн страницы автоматически подстраивается при изменении ширины окна браузера. Поэтому, например, на мобильном устройстве любая многоколоночная страница мгновенно изменяется и становится одноколоночной, что упрощает чтение.
- Pure.css (purecss.io) — нечто большее, чем просто CSS-фреймворк. Эта система позволяет создавать стили CSS, с помощью которых можно формировать не только сетки, но и оформлять кнопки, таблицы, меню и формы. Pure.css — проект компании Yahoo!.
- Foundation (foundation.zurb.com) — другой адаптивный CSS-фреймворк. Эта система более попу-

лярна, но и сложнее, чем большинство других систем модульной верстки. Она прекрасно подходит для создания веб-приложений и интерактивных сайтов, поскольку содержит не только таблицы каскадных стилей, но и JavaScript-код для создания раскрывающихся списков, графических элементов, подсказок и модальных диалоговых окон.

- Bootstrap (getbootstrap.com) — наиболее популярный CSS-фреймворк (этим фактом объясняется схожесть многих сайтов). Созданный компаниями Twitter и Bootstrap, этот фреймворк используется в тысячах сайтов. Он включает в себя модульную сетку и не только, как и сервис Foundation. Фреймворк Bootstrap содержит правила форматирования кнопок, таблиц, предупреждений, значков, ярлыков и даже компонент *jumbotron* (<http://getbootstrap.com/components/#jumbotron>). Он предназначен для создания контента или информации на сайте, который занимает всю ширину контейнера. Кроме того, фреймворк Bootstrap располагает многими компонентами JavaScript, такими как подсказки, карусели и модальные диалоговые окна.

Использование системы модульной верстки Skeleton

В этой главе вы узнаете, как работать со Skeleton — простой, гибкой системой модульной верстки веб-страниц. Вы также изучите несколько дополнительных правил каскадных таблиц стилей, которые призваны помочь с базовым форматированием кнопок, веб-форм и таблиц. Система Skeleton выполняет всю эту работу с помощью 400 строк кода, помещенных в файл небольшого размера. Она создана прежде всего для работы с проектами для мобильных устройств и проста в использовании.

Skeleton — адаптивная система модульной верстки, поскольку при уменьшении ширины окна браузера до 550 пикселей она сворачивает в одну колонку многоколоночную страницу. На экранах смартфонов пространство особенно ценно и отображать текст или любой другой контент в нескольких колонках нецелесообразно — его слишком трудно читать. Поэтому большинство веб-дизайнеров проектов для мобильных устройств для отображения контента используют одну колонку, в которой все элементы `div` расположены один за другим.

Система Skeleton позволяет сверстать сложный сетчатый макет, аккуратно структурировав контент страницы в несколько колонок на планшетных устройствах, ноутбуках и компьютерах. На смартфоне, однако, каскадная таблица стилей Skeleton автоматически удаляет колонки, расположенные рядом друг с другом, преобразуя дизайн страницы в одноколоночный, удобный для чтения. Вам не придется делать ничего; адаптивный дизайн встроен прямо в CSS-файл системы Skeleton.

Чтобы начать работу, посетите сайт Skeleton (getskeleton.com) и нажмите кнопку **Download** (Скачать) (рис. 16.2). Загруженный архив содержит несколько папок и файлов, но все, что вам нужно, — содержимое каталога **CSS**: файл `normalize.css`, который обеспечивает сброс базовых стилей CSS (см. раздел «Устранение конфликтов стилей в браузере» главы 18), чтобы все браузеры отображали HTML-элементы одинаково, и файл `skeleton.css`, содержащий все необходимое, чтобы скомпоновать макет с использованием модульной сетки.

Ниже приведены основные шаги по применению системы модульной верстки Skeleton.

1. Прикрепите CSS-файлы.

Вы можете прикрепить файлы `normalize.css` и `skeleton.css` с помощью следующего кода:

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

ПРИМЕЧАНИЕ

Базовые стили системы Skeleton используют веб-шрифт Google под названием *Raleway*. Это красивый шрифт, который, к сожалению, не поддерживает кириллицу. Поэтому в рамках примеров данной книги он был заменен на аналогичный с поддержкой кириллицы — *Roboto*. Если вы хотите использовать его на своем сайте, вам необходимо добавить ссылку на файлы этого шрифта. Для этого используйте следующий код на своей веб-странице:

```
<link <link href="https://fonts.googleapis.com/css?family=Roboto:400,300,700&subset=latin,Cyrillic"
rel="stylesheet" type="text/css">
```

2. Добавьте контейнеры `div`.

Как упоминалось ранее в этой главе, в модульной сетке контейнеры `div` используются для построения строк и колонок. Страница может иметь только один контейнер, но бывают случаи, когда для создания определенного эффекта необходимо несколько контейнеров (подробнее об этом читайте в следующем разделе). Контейнер в системе Skeleton — это обычный элемент `div` с классом `container`:

```
<div class="container">
```

```
</div>
```

Не помещайте какой-либо HTML-код в контейнер `div`, кроме элементов `div` для строк, как вы это сделаете на следующем шаге.

3. Добавьте элементы `div` для строк.

В каждый из контейнеров вам необходимо добавить одну или несколько строк. Это обычные элементы `div`, вложенные в контейнер `div`. Им присваивается класс `row`:

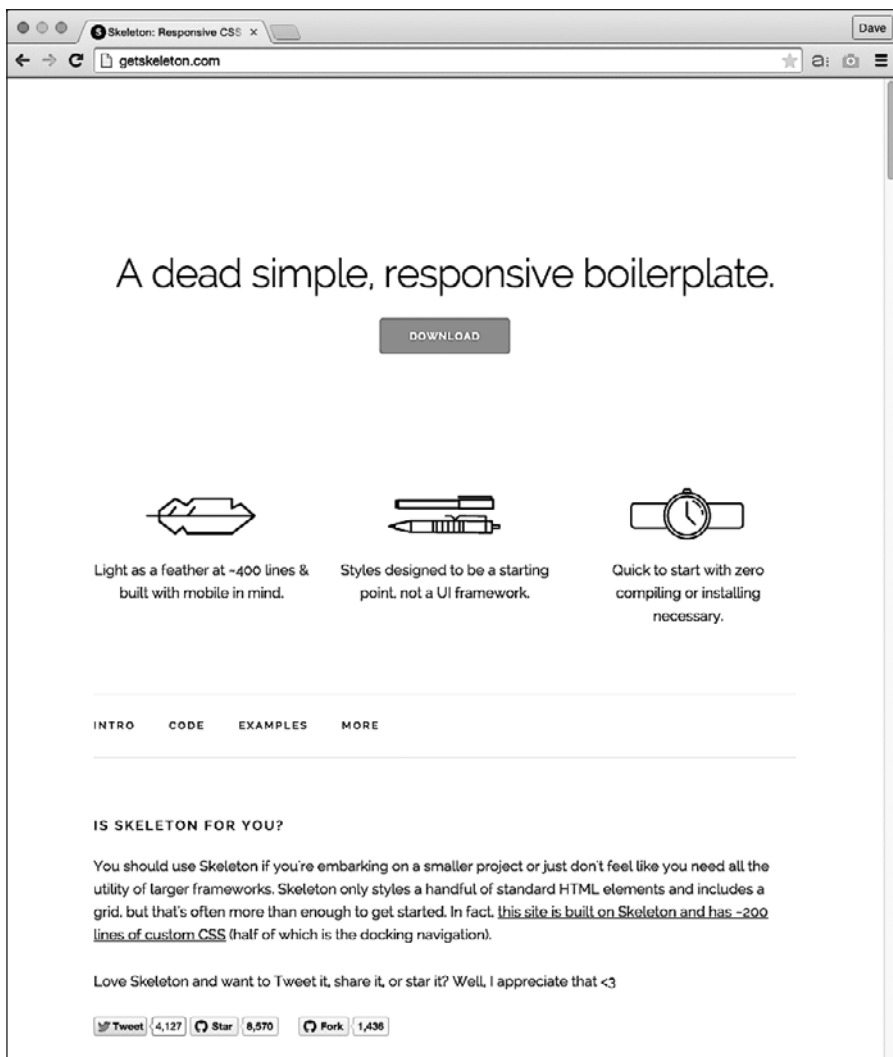


Рис. 16.2. Skeleton — это набор правил CSS, которые предоставляют простую, адаптивную модульную сетку для верстки веб-страниц

```
<div class="container">
  <div class="row">

    </div>
  </div>
```

В один контейнер можно поместить любое количество строк. В самом деле, если вы не используете определенный эффект дизайна, описанный в следующем разделе, можете применять единственный контейнер для страницы, содержащей несколько строк и колонок:

```
<div class="container">
  <div class="row">

    </div>
    <div class="row">

      </div>
      <div class="row">

        </div>
      </div>
    </div>
  </div>
```

4. Добавьте элементы div для колонок.

Система Skeleton основана на модульной сетке, состоящей из 12 столбцов, поэтому каждый из добавленных вами элементов div должен быть не менее одного (что очень мало) или не более 12 (полная ширина контейнера) столбцов в ширину. Кроме того, если вы используете более одной колонки, их общая ширина должна составлять 12 столбцов. Например, если у вас есть три колонки, каждая из них может быть шириной четыре столбца или, к примеру, одна колонка может быть шириной два столбца, в то время как две другие — по пять столбцов.

Вы определяете элементы div как колонки, добавив имя columns в качестве атрибута класса. Затем укажите ширину этой колонки, используя число в диапазоне от 1 до 12. Помните, что имена классов в CSS не могут начинаться с чисел, например 1, 2 или 12, поэтому вам необходимо писать числа словами. Например, чтобы создать три равные по ширине колонки, вы можете добавить три следующих элемента div:

```
<div class="container">
  <div class="row">
    <div class="four columns">

      </div>
    <div class="four columns">

      </div>
    <div class="four columns">

      </div>
    </div>
  </div>
</div>
```

СОВЕТ

Если вы хотите создать одну колонку в строке, не добавляйте элементы div внутри строки. Просто добавьте необходимый контент следующим образом:

```
<div class="row">
  <p>Этот текст заполняет всю ширину контейнера. Получается одна колонка.</p>
</div>
```

5. Добавьте контент в элементы `div` колонок.

Контейнер и элементы `div` строк — структурные элементы, используемые для создания колонок. Контент, например текст, фотографии и видеоролики, находится внутри каждого из элементов `div` колонок.

6. Создайте собственные стили.

Таблица стилей `Skeleton` не позволяет ничего, кроме создания базовой структуры вашего контента, то есть *каркаса*. Чтобы придать странице оформление, вам нужно добавить собственные стили, с помощью которых сайт декорируется цветом, шрифтами, фоновыми изображениями и т. д. Я рекомендую для этих целей создать еще один CSS-файл под именем, к примеру, `custom.css` и связать его с веб-страницей после ссылки на файл `skeleton.css`:

```
<link rel="stylesheet" href="css/normalize.css">
<link rel="stylesheet" href="css/skeleton.css">
<link rel="stylesheet" href="css/custom.css">
```

Система `Skeleton` в плане дизайна в первую очередь отдает предпочтение мобильным устройствам (см. раздел «Медиазапросы» главы 15), поэтому для подгонки дизайна под окна браузеров различной ширины вы будете использовать медиазапросы, начиная с нацеленных на узкий экран смартфона (в подразделе «Приоритет мобильных устройств» раздела «Создание и именование колонок» далее вы узнаете, как сделать это).

Создание и именование колонок

В системе `Skeleton` используется модульная сетка, состоящая из 12 столбцов (рис. 16.3). Вы можете создавать колонки разной ширины, размером от 1 до 12 столбцов. Для создания колонки присвойте два класса элементу `div`. Один класс определяет ширину (`one`, `two`, `three` и т. д.), в то время как другой использует слово `columns`. Имейте в виду, что это не единое имя класса CSS: следует добавить имена двух классов, и они должны быть отделены друг от друга пробелом. Каждое имя относится к соответствующему правилу каскадной таблицы стилей `Skeleton`.

Обратите внимание на следующие два примера. Скажем, вы хотите создать строку с двумя колонками одинаковой ширины. Все, что нужно сделать, — это добавить два элемента `div`, каждый шириной шесть столбцов, внутри элемента `div` с классом `row`, вложенного в контейнер `div` с классом `container`, как это показано ниже:

```
<div class="container">
  <div class="row">
    <div class="six columns">

    </div>
    <div class="six columns">

  </div>
</div>
</div>
```

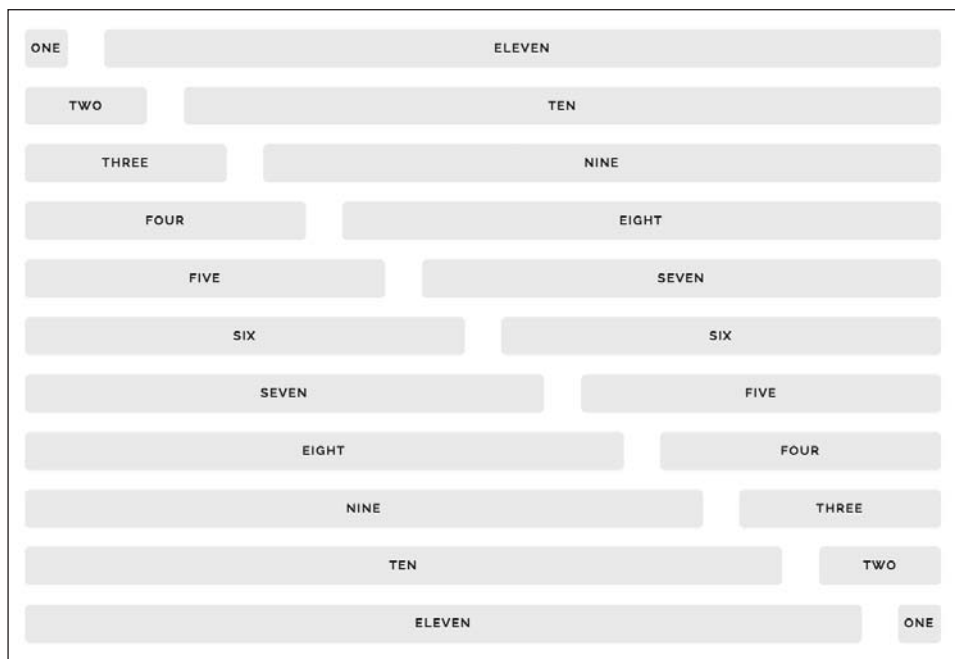


Рис. 16.3. Система модульной верстки Skeleton использует сетку, состоящую из 12 столбцов, которая позволяет создавать комбинации колонок различной ширины. Как видно из рисунка, ширина колонки в один столбец крайне мала, поэтому вряд ли вы когда-нибудь создадите колонку такой ширины. Но поскольку столбцов 12, вы можете создавать различные комбинации колонок разной ширины

Чтобы создать четыре колонки одинаковой ширины, добавьте четыре элемента `div`, по три столбца в ширину каждый:

```
<div class="container">
  <div class="row">
    <div class="three columns">

  </div>
  <div class="three columns">

  </div>
  <div class="three columns">

  </div>
  <div class="three columns">

  </div>
</div>
```

Колонки не обязательно должны иметь одинаковую ширину. Тем не менее необходимо убедиться, что сумма всех колонок в строке равна 12 столбцам. Например,

чтобы в строке было создано две колонки, одна из которых имеет ширину четыре столбца, а другая — восемь, код должен выглядеть следующим образом:

```
<div class="container">
  <div class="row">
    <div class="four columns">

    </div>
    <div class="eight columns">

    </div>
  </div>
</div>
```

Одним очень полезным аспектом системы Skeleton является то, что она автоматически (и точно) вычисляет средники между несколькими колонками. То есть вам не нужно вычислять, какое пространство должно быть между двумя колонками. Если в вашей строке только одна колонка, таблица стилей Skeleton не добавит средник; если колонки две, она добавит один средник, чтобы разделить их; а если колонок шесть — будет добавлено пять средников, размеры которых будут вычислены автоматически в CSS-файле системы Skeleton!

СОВЕТ

Для создания колонок шириной в половину, одну треть и две трети от ширины контейнера система Skeleton использует сокращенные имена классов. Для этого укажите значения `half`, `one-third` или `two-thirds` соответственно рядом с именем класса `column`. Например, чтобы создать две колонки, ширина одной из которых равна одной трети от ширины контейнера, а второй — две трети, используйте следующий код:

```
<div class="row">
  <div class="one-third column">

  </div>
  <div class="two-thirds column">

  </div>
</div>
```

Создание разделов во всю ширину окна браузера

Контейнер `div`, добавленный на страницу, содержит строки и колонки. CSS-код таблицы Skeleton не разворачивает стиль `container` так, чтобы контейнер заполнял окно браузера в устройствах просмотра. Ширина контейнера определяется посредством медиазапросов, но никогда не распространяется от края до края окна. В некоторых случаях это допустимо, но, если вы хотите залить фон контейнера цветом или изображением, ваш дизайн будет выглядеть, мягко говоря, не очень хорошо (рис. 16.4, *вверху*). В этом случае необходимо развернуть фон до краев окна браузера (см. рис. 16.4, *внизу*).

К счастью, это очень просто сделать. Поскольку ширина любого блочного элемента, такого как `div`, по умолчанию составляет 100 %, вам необходимо лишь обернуть элементом `div` контейнер `div` и установить цвет фона для внешнего `div` (обертки).

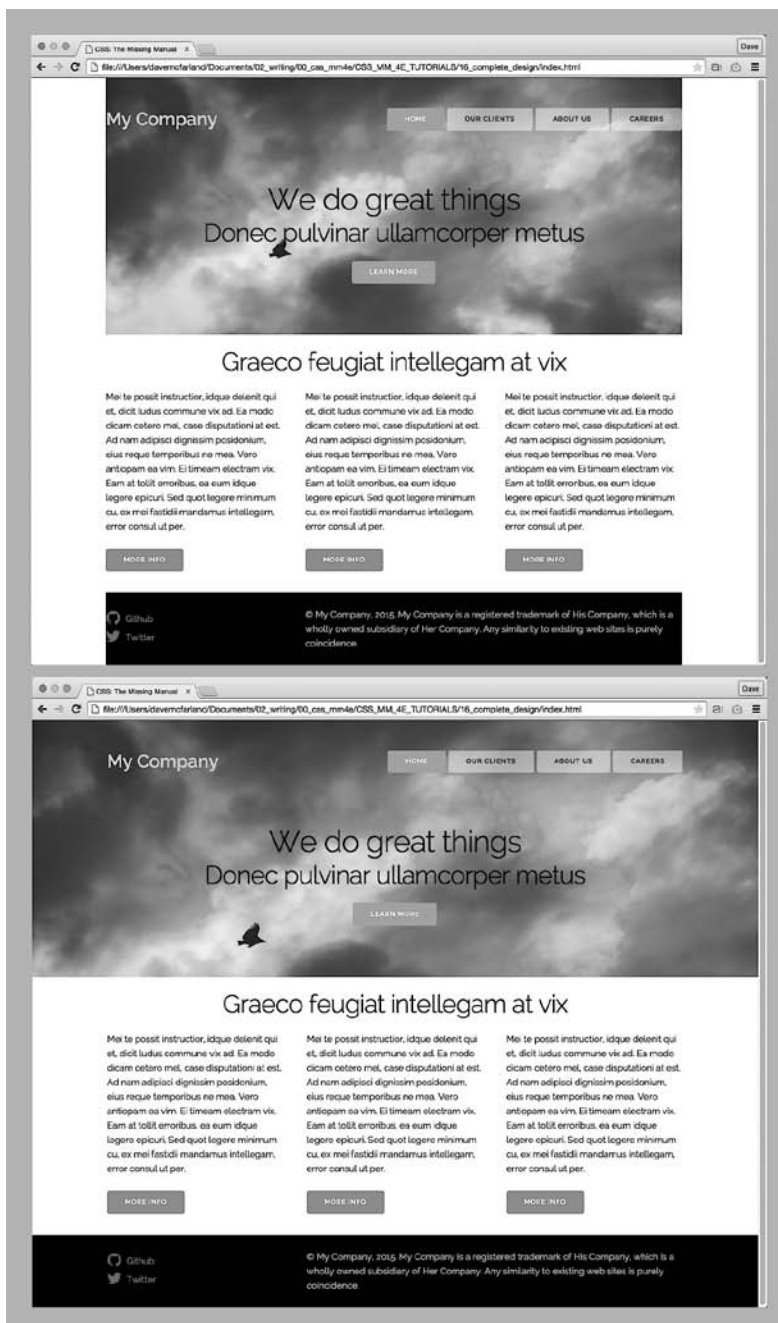


Рис. 16.4. Класс container таблицы стилей Skeleton не распространяется от края до края окна браузера (вверху). Если вы назначили фоновое изображение или цвет одному из таких контейнеров, назначение не будет распространяться за рамки контейнера. В большинстве случаев страница выглядит лучше, если фоновое изображение или цвет распространяются до краев окна браузера (внизу)

Например, на нижнем изображении на рис. 16.4 в верхней области страницы расположена фотография, которая заполняет область заголовка между краями окна браузера. Это достигается с помощью следующего HTML-кода для данной области:

```
<div class="section header">
  <div class="container">
    <div class="row">
      <div class="three columns">
        <p class="logo">My Company</p>
      </div>
      <div class="nav nine columns">
        <a class="button button-primary" href="#">Home</a>
        <a class="button" href="#">Our Clients</a>
        <a class="button" href="#">About Us</a>
        <a class="button" href="#">Careers</a>
      </div>
    </div>
    <div class="row action">
      <h1>We do great things</h1>
      <h2>Donec pulvinar ullamcorper metus</h2>
      <a href="#" class="button button-primary">Learn More</a>
    </div>
  </div>
</div>
```

Важной частью является внешний элемент `div`. Он окружает контейнер `div` (строки 2 и 19). Присваивая имя класса этому элементу `div`, вы можете легко установить фоновое изображение с помощью CSS:

```
.header {
  background-image: url(../imgs/header.jpg);
  background-size: cover;
}
```

Теперь, поскольку ширина внешнего элемента `div` всегда равна 100 %, его фон будет всегда заполнять всю ширину окна браузера.

Как уже упоминалось, для всей веб-страницы, возможно, потребуется только один контейнер `div`. Тем не менее вы можете добавлять *различные* фоны для различных строк, тем самым заполняя всю ширину окна.

Например, на нижнем изображении на рис. 16.4 фотография заполняет верхнюю часть страницы, а черная полоса — нижнюю. К различным частям страницы применены два разных фона. В этом случае вам необходимо использовать несколько контейнеров: один для верхней части (фотография), один для средней части (белая область) и один — для нижнего колонтитула (черная полоса). Кроме того, каждый из этих контейнеров должен быть обернут собственным элементом `div`. Эти `div`-элементы заполняют ширину окна браузера, и вы можете настроить стиль каждого из них по отдельности. В практикуме в конце этой главы вы увидите пример создания фона в полную ширину.

Форматирование кнопок

Несмотря на то что Skeleton, по сути, модульная сетка, она позволяет использовать пару забавных, привлекательных стилей для форматирования других типов элементов страницы. В частности, Skeleton создает кнопки с отличным оформлением (рис. 16.5).

CSS-файл Skeleton автоматически форматирует определенные HTML-элементы, чтобы они выглядели как кнопки, изображенные на рис. 16.5. Например, он форматирует HTML-элемент `button`, а также элементы формы типа `submit` или `button` (см. белые кнопки, изображенные в верхней части рис. 16.5).

К примеру, на вашей веб-странице находится следующий HTML-код:

```
<button>Текст кнопки</button>
```

CSS-файл Skeleton автоматически придаст им вид белой кнопки со скругленными углами и серой границей. Теперь, предположим, вы хотите придать такой же стиль другим HTML-элементам. Например, чтобы ряд ссылок, образующих панель навигации, выглядел как кнопки Skeleton. Для этого добавьте имя класса `button` к ссылке:

```
<a href="index.html" class="button">Главная</a>
```

Таблица стилей Skeleton содержит еще один класс, `.button-primary`, который создает синюю подсветку кнопки. Для ее реализации добавьте следующий класс в HTML-код:

```
<button class="button-primary">Текст кнопки</button>
```

Если вы хотите превратить обычную ссылку в подсвеченную кнопку, добавьте два класса: один из них преобразует ссылку в кнопку, а другой подсветит ее:

```
<a href="index.html" class="button button-primary">Главная</a>
```

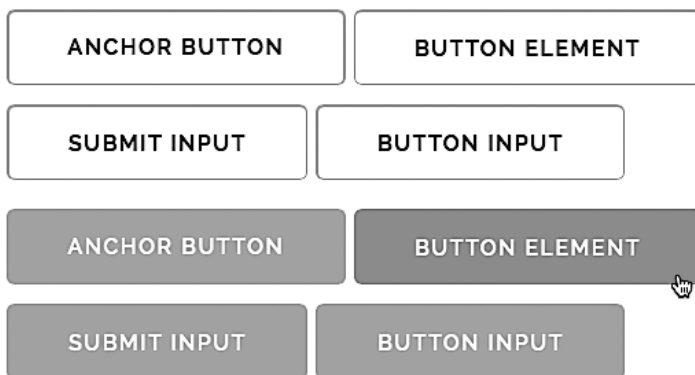


Рис. 16.5. С помощью системы Skeleton очень легко превратить ссылки и элементы формы в простые, привлекательные кнопки

Приоритет мобильных устройств

Таблица стилей Skeleton отдает предпочтение мобильным устройствам. Стратегия Mobile First, о которой вы прочитали в разделе «Медиазапросы» в главе 15, предполагает применение дизайна, который в первую очередь выглядит хорошо на небольших экранах мобильных устройств, таких как смартфоны. Используйте медиазапросы, а затем добавляйте стили, которые улучшают внешний вид страницы на экранах с разным разрешением по горизонтали. Каждый медиазапрос определяет более высокое разрешение крана по горизонтали, а вы добавляете стили для улучшения дизайна для каждой из этих точек останова (экранное разрешение по горизонтали).

Стратегия Mobile First предполагает использование свойства `min-width` в правиле `@media`. Другими словами, каждый медиазапрос применяет разные стили, если экран обладает тем или иным минимальным разрешением по горизонтали. Например, взгляните на следующий медиазапрос:

```
@media (min-width: 400px) {  
  
}
```

Исходя из него добавленные в данный медиазапрос стили будут применяться только в тех случаях, если окно браузера имеет размер не менее 400 (и более) пикселей в ширину. Если ширина окна или разрешение экрана, к примеру, 320 пикселей, стили внутри этого медиазапроса не будут применены к странице. Подход, основанный на приоритете для мобильных устройств, заключается в применении нескольких правил `@media` с разным значением свойства `min-width`.

При использовании системы Skeleton для создания дизайна по стратегии Mobile First рекомендуется начать с отдельной таблицы стилей. Присвойте ей имя, например `custom.css` или `site.css`, и прикрепите к веб-странице после CSS-файлов системы Skeleton:

```
<link rel="stylesheet" href="css/normalize.css">  
<link rel="stylesheet" href="css/skeleton.css">  
<link rel="stylesheet" href="css/custom.css">
```

Вы сможете добавлять правила в этот CSS-файл, чтобы определить внешний вид сайта, изменив цвета, шрифты, фон и т. д. Для создания дизайна по стратегии Mobile First вы должны начать с определенных базовых медиазапросов. Разработчик системы Skeleton рекомендует настраивать медиазапросы для ряда различных точек останова, например:

```
/* Запросы Mobile First */  
/* стили для любой ширины */  
  
/* стили для ширины крупнее мобильных */  
/* стили для ширины 400 пикселей и выше */  
@media (min-width: 400px) {  
  
}  
  
/* стили для ширины крупнее фэблетов */  
/* стили для ширины 550 пикселей и выше */
```

```
@media (min-width: 550px) {  
  
}  
/* стили для ширины крупнее планшетов */  
/* стили для ширины 750 пикселей и выше */  
@media (min-width: 750px) {  
  
}  
/* стили для ширины крупнее компьютерных мониторов */  
/* стили для ширины 1000 пикселей и выше */  
@media (min-width: 1000px) {  
  
}  
/* стили для ширины крупнее компьютерных HD-мониторов */  
/* стили для ширины 1200 пикселей и выше */  
@media (min-width: 1200px) {  
  
}
```

В дизайне по стратегии Mobile First стили, которые применяются для всех устройств (мобильных или нет), добавляются *вне* медиазапросов. Например, если вы хотите применить тот же шрифт, цвет шрифта, цвет фона для элементов страницы независимо от ширины окна браузера, добавьте эти правила CSS здесь.

Кроме того, здесь вы должны добавить стили, которые форматируют ваш сайт для просмотра на мобильном устройстве. Вы можете использовать панель **Developer Tools** (Инструменты разработчика) в браузере Chrome, чтобы просмотреть, как страница будет выглядеть на мобильном устройстве.

После того как мобильная версия будет выглядеть должным образом, расширьте окно браузера (или используйте панель **Developer Tools** (Инструменты разработчика) в браузере Chrome), чтобы увидеть, как ваша страница будет выглядеть в первой точке останова — при ширине экрана, равной 400 пикселям. Если при этой ширине страница выглядит прекрасно, вы можете расширить окно браузера до следующей точки останова (550 пикселей).

Если страница выглядит не очень хорошо, возможно, на ней не хватает воздуха вокруг элементов. В этом случае добавьте новые стили в этой точке останова, пока страница не будет выглядеть хорошо. Помните, любое дополнение, которое вы делаете в медиазапросе, не будет применяться к более ранней точке останова. Так, например, если вы добавите стиль, который увеличивает отступ по периметру заголовков, в медиазапросе со значением свойства `min-width`, равным 550 пикселям, отступы не будут добавлены к заголовкам при размере окна браузера меньше 550 пикселей в ширину. Другими словами, изменения, внесенные в этот медиазапрос, не повлияют на дизайн страницы при ее просмотре на экране смартфона.

Вы можете продолжить процесс просмотра внешнего вида сайта в каждой точке останова, добавляя нужные стили в соответствующие медиазапросы, чтобы улучшить внешний вид страницы при конкретной ширине окна браузера. Ранее приведенный код содержит пять медиазапросов с пятью различными точками останова, но вовсе не обязательно добавлять стили в каждой из них. Вероятно, вам только потребуется настроить внешний вид элементов страницы

для нескольких различных значений ширины окна браузера (разрешения экрана по горизонтали). Кроме того, вы можете обнаружить, что значения свойства `min-width`, используемые в этом коде, не работают с конкретным дизайном. Например, дизайн вашей страницы может выглядеть отлично вплоть до ширины 650 пикселей, но при дальнейшем увеличении элементы страницы не вписываются или выглядят плохо. В этом случае измените значение свойства `min-width` для конкретной точки останова и добавьте нужные стили внутри этого медиазапроса (например, измените значение 550px на 650px). Вполне возможно, вам и не понадобятся все эти медиазапросы, приведенные в коде, поэтому вы можете удалить их из таблицы стилей.

Лучший способ изучить дизайн по стратегии Mobile First — создать его самостоятельно. Сейчас вы получите такую возможность.

Практикум: использование системы модульной верстки

Этот практикум продемонстрирует, как применить модульную сетку к веб-странице с помощью системы модульной верстки Skeleton. Вы начнете с некоторых основных элементов HTML-кода, примените необходимые CSS-файлы, добавите и структурируете HTML-код, чтобы создать сетку, и, наконец, напишете собственные правила CSS, чтобы придать странице отличный вид даже при ее просмотре на экранах мобильных устройств.

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 16.

Добавление сетки

Сначала вам необходимо прикрепить CSS-файлы и добавить некоторый HTML-код, чтобы создать контейнер таблицы со строками и столбцами.

1. В своем редакторе HTML-кода откройте файл `index.html`, расположенный в папке 16.

Это простой HTML-файл. Он содержит разделы заголовка и тела, в котором еще нет HTML-кода. Однако прежде, чем добавить HTML-код, добавьте ссылки на несколько CSS-файлов. В файле уже есть ссылка на веб-шрифт Google, и вам необходимо прикрепить файлы таблицы стилей Skeleton.

2. В пустой строке сразу после закрывающего тега `</head>` добавьте следующие три строки кода:

```
<link rel="stylesheet" href="css/normalize.css">
<link rel="stylesheet" href="css/skeleton.css">
<link rel="stylesheet" href="css/custom.css">
```

Первая строка кода загружает файл `normalize.css`. Это очень популярный файл сброса стилей CSS (см. раздел «Устранение конфликтов стилей в браузере» главы 18), который используется во многих проектах. Второй файл — это основной файл системы модульной верстки Skeleton — `skeleton.css`, который содержит CSS-код модульной сетки. Последний файл — `custom.css` — по сути, пуст. Он содержит только несколько медиазапросов без самих стилей, которые применяются для подгонки страниц сайта под экраны различных размеров. Вы будете использовать этот файл для добавления стилей и настройки внешнего вида страницы.

Далее вы добавите контейнер Skeleton.

3. Сразу после комментария `<!-- основной раздел -->` добавьте следующий HTML-код:

```
<!-- основной раздел -->
<div class="container">

</div>
```

Как уже говорилось, таблица стилей Skeleton использует имя класса `container`, чтобы определить элемент `div`, который содержит добавленные вами строки на странице. На странице может использоваться как один, так и любое другое количество контейнеров `div` для строк. Как вы скоро убедитесь, при использовании более чем одного контейнера всегда есть преимущество.

Добавим первую строку.

4. После только что добавленного элемента `div` вставьте еще два элемента `div`, чтобы создать две строки (добавленный текст выделен полужирным шрифтом):

```
<!-- основной раздел -->
<div class="container">
  <div class="row">

    </div>
    <div class="row">

    </div>
  </div>
</div>
```

Добавленные элементы `div` создают две строки. В первую строку вы добавите две колонки — одну для названия сайта, а вторую — для панели навигации.

5. В первом элементе `div` с классом `row` добавьте следующий HTML-код:

```
<div class="four columns">
  <p class="logo">Моя компания</p>
</div>
<div class="eight columns nav">

</div>
```

Система модульной верстки Skeleton использует модульную сетку в 12 столбцов. Вы можете использовать эти столбцы, чтобы сделать отдельные колонки. В данном

примере вы добавили две колонки: ширина одной из них равна четырем столбцам, а второй — восьми. Визуально первая колонка занимает одну треть, или около 33 % всей ширины контейнера, а вторая — две трети, или около 66 %.

Вторая колонка шире, потому что в ней будет находиться панель навигации. На самом деле обратите внимание, что к данному элементу добавлен класс `nav`. Это не является требованием каркаса `Skeleton`. Вы добавили его сейчас, чтобы использовать позже для форматирования панели навигации.

6. Откройте файл `01-nav.html`. Скопируйте содержащийся в нем HTML-код и вставьте его в элемент `div` с классом `eight columns nav` файла `index.html`, чтобы HTML-код выглядел следующим образом (добавленный текст выделен полужирным шрифтом):

```
<div class="eight columns nav">
  <a href="#">Главная</a>
  <a href="#">Клиенты</a>
  <a href="#">0 нас</a>
  <a href="#">Вакансии</a>
</div>
```

Это простые ссылки. Тем не менее таблица стилей `Skeleton` включает в себя несколько очень красивых стилей для форматирования любого элемента в качестве кнопки. Для этого вы добавите имя класса к элементам `a`.

7. Добавьте атрибут `class` к каждой ссылке и присвойте ему имя `button`:

```
<div class="eight columns nav">
  <a class="button" href="#">Главная</a>
  <a class="button" href="#">Клиенты</a>
  <a class="button" href="#">0 нас</a>
  <a class="button" href="#">Вакансии</a>
</div>
```

Система `Skeleton` также включает в себя специальный класс с именем `button-primary`, который придает кнопке особый вид. Это отличный способ выделить ссылку на текущей странице среди остальных ссылок, чтобы посетители видели, на какой странице они находятся. Страница, над которой вы работаете, является главной, поэтому вы добавите специальный класс к ссылке «Главная».

8. Добавьте имя класса `button-primary` к первому элементу `a`:

```
<div class="eight columns nav">
  <a class="button button-primary" href="#">Главная</a>
  <a class="button" href="#">Клиенты</a>
  <a class="button" href="#">0 нас</a>
  <a class="button" href="#">Вакансии</a>
</div>
```

Осталась последняя строка, к которой необходимо добавить контент. Это просто, поскольку она содержит только одну колонку, заполняющую контейнер по всей ширине.

9. Откройте файл `02-action.html`. Скопируйте HTML-код, вернитесь к файлу `index.html` и вставьте его во второй элемент `div` с классом `row`, находящийся внутри

элемента `div` с классом `container`. Результат операции должен выглядеть следующим образом (добавленный текст выделен полужирным шрифтом):

```
<!-- основной раздел -->
<div class="container">
  <div class="row">
    <div class="four columns">
      <p class="logo">Моя компания</p>
    </div>
    <div class="eight columns nav">
      <a class="button button-primary" href="#">Главная</a>
      <a class="button" href="#">Клиенты</a>
      <a class="button" href="#">0 нас</a>
      <a class="button" href="#">Вакансии</a>
    </div>
  </div>
  <div class="row">
    <h1>Мы творим великие дела</h1>
    <h2>Donec pulvinar ullamcorper metus</h2>
    <a href="#" class="button button-primary">Подробнее</a>
  </div>
</div>
```

Содержимое этой строки будет распространяться на всю ширину контейнера, другими словами, будет создана только одна колонка. В этом случае вы не добавляли дополнительные элементы `div`; вы просто вставили HTML-код внутри элемента `div` с классом `row`.

10. Добавьте класс `action` ко второму элементу `div` с классом `row`:

```
<div class="row action">
  <h1>Мы творим великие дела</h1>
  <h2>Donec pulvinar ullamcorper metus</h2>
  <a href="#" class="button button-primary">Подробнее</a>
</div>
```

Вы будете использовать это имя класса позже, при форматировании этой области страницы. Имя `action` не имеет ничего общего с таблицей стилей Skeleton — это лишь имя класса, которое вы можете использовать для форматирования данной области страницы.

11. Сохраните файл `index.html` и просмотрите его в браузере.

Страница должна выглядеть так, как показано на рис. 16.6. На данный момент не похоже, что для нее используется разрекламированная модульная сетка. Но мы это исправим. Добавим еще одну строку и три колонки. Чтобы выполнить эту операцию быстрее, базовый HTML-код контента для этой страницы сохранен в отдельном файле.

12. Откройте файл `03-info.html`. Скопируйте HTML-код, вернитесь к файлу `index.html` и вставьте его после комментария `<!-- раздел контента -->`.

Это базовый набор вложенных элементов `div`. Один из элементов `div` представляет собой контейнер для строк, а два элемента `div` внутри него — строки. В первой



Рис. 16.6. Таблица стилей Skeleton предоставляет не только модульную сетку, но и некоторые базовые текстовые стили и красивые кнопки

строке содержится только элемент `h2`, который будет занимать одну колонку во всю ширину контейнера. Во второй строке находятся три дополнительных элемента `div`, которые представляют собой три колонки.

HTML-код еще не содержит имен классов Skeleton — это секретный ингредиент, который применяет модульную сетку. Сейчас вы его добавите.

13. Добавьте код `class="container"` к первому элементу `div` в HTML-код, который вы только что вставили на страницу:

```
<!-- раздел контента -->
<div class="container">
```

Этот код применяет стиль контейнера Skeleton к элементу, создавая пространство для добавления строк. Далее вы добавите несколько строк.

14. Добавьте код `class="row"` к следующим двум элементам `div` внутри контейнера, как это показано в следующем примере (добавленный код выделен полужирным шрифтом):

```
<div class="container">
  <div class="row">
    <h2>Graeco feugiat intellegam at vix</h2>
  </div>
  <div class="row">
```

Теперь, когда вы создали строки, во второй строке необходимо создать три колонки. Вы можете сделать это, добавив атрибут `class` с двумя именами классов: первое имя — количество столбцов, которое занимает колонка по ширине, а второе — `columns`.

15. Добавьте код `class="four columns"` к каждому элементу `div` внутри второго `div` с классом `row` в раздел контента страницы. Кроме этого, добавьте класс `button` к трем ссылкам в каждой из колонок. Законченный HTML-код для раздела контента страницы должен выглядеть следующим образом:

```
<!-- раздел контента -->
<div class="container">
  <div class="row">
    <h2>Graeco feugiat intellegam at vix</h2>
```

```

</div>
<div class="row">
  <div class="four columns">
    <p>Mei te possit instruction...</p>
    <p><a href="#" class="button">Дополнительно</a></p>
  </div>
  <div class="four columns">
    <p>Mei te possit instruction...</p>
    <p><a href="#" class="button">Дополнительно</a></p>
  </div>
  <div class="four columns">
    <p>Mei te possit instruction...</p>
    <p><a href="#" class="button">Дополнительно</a></p>
  </div>
</div>
</div>

```

(Приведенный HTML-код сокращен и не включает «бред» на латинице в каждой из колонок.) Наконец, вы добавите нижний колонтитул на страницу.

16. Откройте файл 04-footer.html. Скопируйте HTML-код, вернитесь к файлу index.html и вставьте его после комментария `<!-- колонтитул -->`.

Этот HTML-код содержит соответствующие классы Skeleton (наверняка вы уже знаете, как работает таблица стилей Skeleton, и практиковаться в ее использовании больше нет смысла). HTML-код, который вы только что добавили, вставляет контейнер с одной строкой и двумя колонками. Ширина одной из них равна восьми столбцам, а второй — четырем столбцам.

17. Сохраните файл index.html и просмотрите его в браузере.

Страница должна выглядеть так, как показано на рис. 16.7. Макет сверстан: в разделе заголовка есть две колонки (для логотипа и панели навигации), три колонки основного контента и две колонки для нижнего колонтитула.

Если вы измените размер окна браузера, сделав его слишком узким, колонки разместятся друг за другом. Так проявляется гибкость системы модульной верстки Skeleton в действии. В ней используются медиазапросы, о которых вы читали в предыдущей главе, позволяющие превратить контент страницы в одноколоночный дизайн для мобильных устройств.

Создание стилей

С помощью Skeleton легко сверстать макет страницы, но для придания ей уникальности нужно добавить дополнительные каскадные таблицы стилей. В этом разделе практикума вы добавите несколько штрихов к дизайну страницы.

1. В редакторе HTML-кода откройте файл custom.css, расположенный в папке 16\css.

По большому счету, этот CSS-файл не содержит стилей. В нем находятся только медиазапросы, предназначенные для устройств с экранами с разным разрешением по горизонтали, для которых вы, возможно, захотите изменить размер



Рис. 16.7. С помощью правильно структурированного HTML-кода, нескольких имен классов и CSS-файла системы модульной верстки Skeleton легко создавать многоколоночные дизайны страниц

и оформление элементов страницы. Например, скорее всего, вы захотите, чтобы при просмотре страницы с помощью смартфона размер заголовка уменьшился, а лишние поля и отступы были удалены.

Для начала посмотрим, что произойдет при добавлении фонового цвета в контейнер.

2. Добавьте следующий стиль сразу после комментария `/* стили для любой ширины */`:

```
/* запросы mobile first */
/* стили для любой ширины */
.container {
  background-color: pink;
}
```

Как вы, возможно, помните, наша страница содержит несколько элементов `div` с классом `container`. Это элементы, которые содержат строки.

3. Сохраните файл `custom.css` и просмотрите файл `index.html` в браузере.

Страница должна выглядеть так, как показано на рис. 16.8. Обратите внимание, что фон розового цвета ограничен по ширине контента и выровнен по центру

окна браузера. Если вы взглянете на окончательный дизайн на рис. 16.4, то увидите, что в области заголовка в качестве фона используется фотография, которая должна отображаться по всей ширине окна браузера. Контейнеры Skeleton не делают этого, поэтому вам нужно добавить еще один элемент `div`, который будет распространяться на всю ширину окна браузера.

4. Откройте файл `index.html` в редакторе HTML-кода. Найдите комментарий `<!-- основной раздел -->` и добавьте элемент `div` сразу после него.

```
<!-- основной раздел -->
<div class="section header">
  <div class="container">
```

Вы только что «обернули» контейнер элементом `div`. Кроме того, вы присвоили ему два имени класса — `section` и `header`, что позволяет форматировать эти области отдельно от других областей страницы. Сейчас необходимо закрыть элемент `div`.

5. Добавьте закрывающий тег `</div>` перед комментарием `<!-- раздел контента -->`:

```
</div>
<!-- раздел контента -->
```

Теперь новый элемент `div` обернут вокруг контейнера. По умолчанию, как и все блочные элементы, `div` имеет ширину 100 %. То есть он растягивается, заполняя свой родительский элемент, который в данном случае является телом страницы. Поэтому новый элемент `div` заполнит окно браузера. Теперь вы можете добавить фотографию, которая заполнит окно браузера по всей ширине, но выйдет за пределы области заголовка.

6. Сохраните файл `index.html` и вернитесь к файлу `custom.css` в редакторе HTML-кода. Удалите правило `.container`, добавленное в шаге 2, и замените его следующим кодом:

```
/* запросы mobile first */
/* стили для любой ширины */
.header {
  padding: 50px 0 70px 0;
  background-image: url(../imgs/header.jpg);
  background-size: cover;
}
```

Свойство `padding` добавляет небольшой отступ в верхней и нижней областях страницы. Свойство `background-size` масштабирует изображение, добавленное с помощью свойства `background-image`, всегда заполняя фон. Если вы сейчас сохраните файлы и просмотрите файл `index.html`, то заметите, что изображение распространяется по всей ширине окна браузера, но располагается только в области заголовка.

Теперь вы можете отформатировать область заголовка.

7. Добавьте стиль для логотипа сразу после стиля `.header`:

```
.logo {
  font-weight: 600;
```

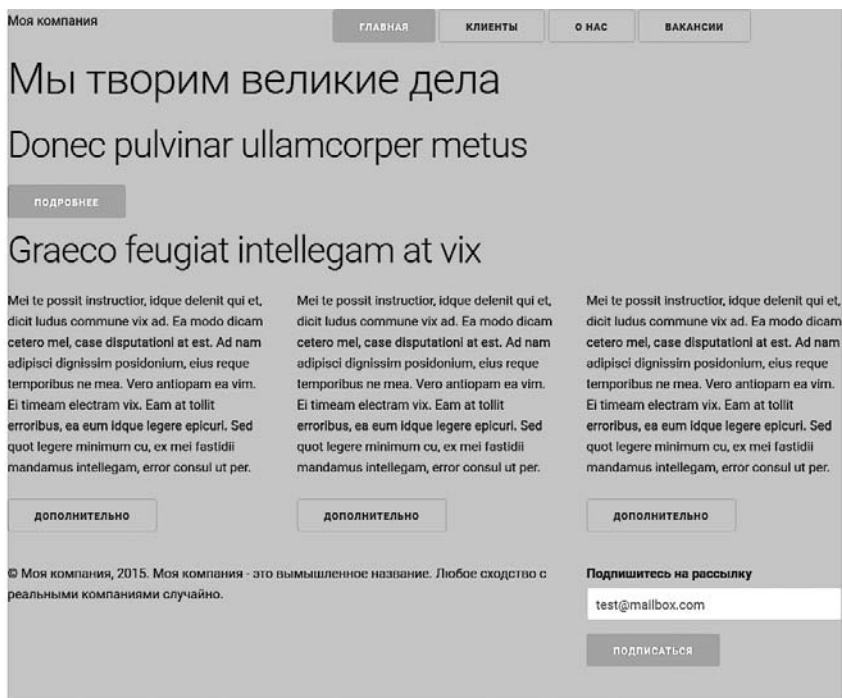


Рис. 16.8. Класс `container` таблицы стилей `Skeleton` не заполняет все окно браузера по ширине. Это проблема, если вы используете фоновый цвет или изображение

```
color: rgb(255, 209, 143);
}
```

Эта страница использует веб-шрифт Google под названием `Roboto`, который включает в себя три различных начертания. Как упоминалось, вы можете использовать числа, чтобы указать различные стили шрифта, начиная с очень тонких и заканчивая очень толстыми (700 — это полужирная версия шрифта). Далее вы выравниваете элементы навигации по правому краю страницы и улучшаете внешний вид кнопок, расположенных поверх фонового изображения.

- Добавьте следующий код после стиля `.logo`:

```
.nav {
  text-align: right;
}
.button {
  background-color: rgba(255,255,255,.5);
}
.button:hover {
  background-color: rgba(255,255,255,.3);
}
```

Стиль `.nav` форматирует колонку с кнопками навигации. В шаге 5 в предыдущем подразделе вы добавили имя `nav` в класс элемента `div`. Два других стиля

изменяют внешний вид кнопки так, чтобы они лучше выделялись на фоне изображения.

Наконец, вы выравниваете по центру текст и кнопку под областью навигации.

9. Добавьте следующий код под только что добавленными стилями кнопок:

```
.action {
  text-align: center;
  padding-top: 37px;
}
.action h1 {
  margin: 0;
}
.action h2 {
  margin: 0 0 20px 0;
}
```

В шаге 10 ранее вы добавили класс `action`. Сейчас вы выравнивали контент в этой области и удалили воздух вокруг заголовков.

10. Сохраните файл и просмотрите `index.html` в браузере.

Страница должна выглядеть так, как показано на рис. 16.9.

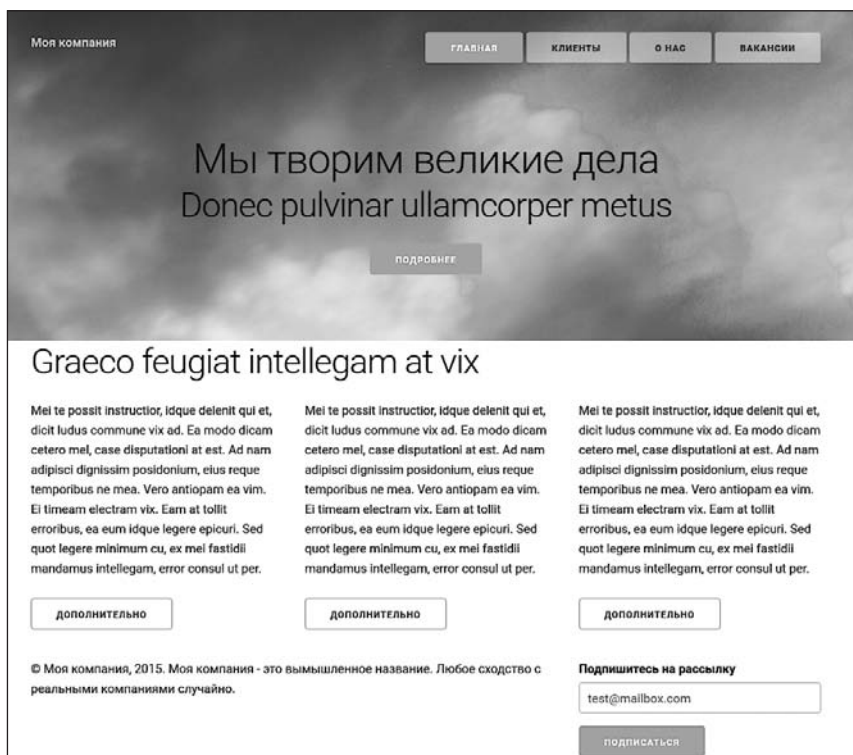


Рис. 16.9. Добавленное изображение и несколько стилей образуют оболочку для каркаса Skeleton

Адаптация под мобильные устройства

Таблица стилей Skeleton в первую очередь предназначена для мобильных устройств. Это значит, что вы начинаете с дизайна для мобильных устройств с небольшим экраном, таких как смартфоны. Затем вы добавите стили внутри нескольких медиа-запросов, которые предназначены для устройств с экранами с более высоким разрешением по горизонтали. Каждый набор стилей предыдущего запроса применяется к текущему запросу и всем остальным запросам с более высокими значениями свойства `min-width`.

Задумайтесь об усовершенствованиях, которые можно сделать на страницах для посетителей, использующих смартфоны. Рисунок 16.10 демонстрирует, как текущая страница будет выглядеть на смартфоне iPhone 5: на ней очень много пространства в верхней части экрана. Вы можете исправить стиль логотипа, чтобы разместить название компании в верхней части экрана, и исправить кнопки навигации, чтобы они лучше соответствовали текущему размеру экрана.

1. В редакторе HTML-кода откройте файл `custom.css`, расположенный в папке `16\css`.

Стили, добавленные в файл ранее в этом уроке, находятся в верхней части файла, перед медиазапросами. В соответствии со стратегией Mobile First эти стили будут применяться, независимо от ширины окна браузера, как на малых экранах смартфонов, так и на очень широких компьютерных мониторах.

Во-первых, вы измените стиль `.logo` так, чтобы дизайн лучше смотрелся на небольшом экране смартфона.

2. Найдите стиль `.logo` и добавьте к нему семь новых строк кода (добавленный код выделен полужирным шрифтом):

```
.logo {  
  font-weight: 600;  
  color: rgb(255, 209, 143);  
  background-color: black;  
  position: fixed;  
  top: 0;  
  left: 0;  
  right: 0;  
  padding-left: 10px;  
  z-index: 100;  
}
```

Свойство `background-color` визуально выделяет название компании. Значения `position`, `top`, `left` и `right` фиксируют этот элемент в верхней части окна браузера. Свойство `padding-left` добавляет небольшой отступ в левой части экрана. И наконец, свойство `z-index` гарантирует, что при прокрутке страницы вниз логотип останется поверх прочего контента страницы (рис. 16.11, *вверху слева*).

Кнопки навигации будут выглядеть лучше, если они будут одинаковой ширины, а свободного пространства между ними останется меньше.

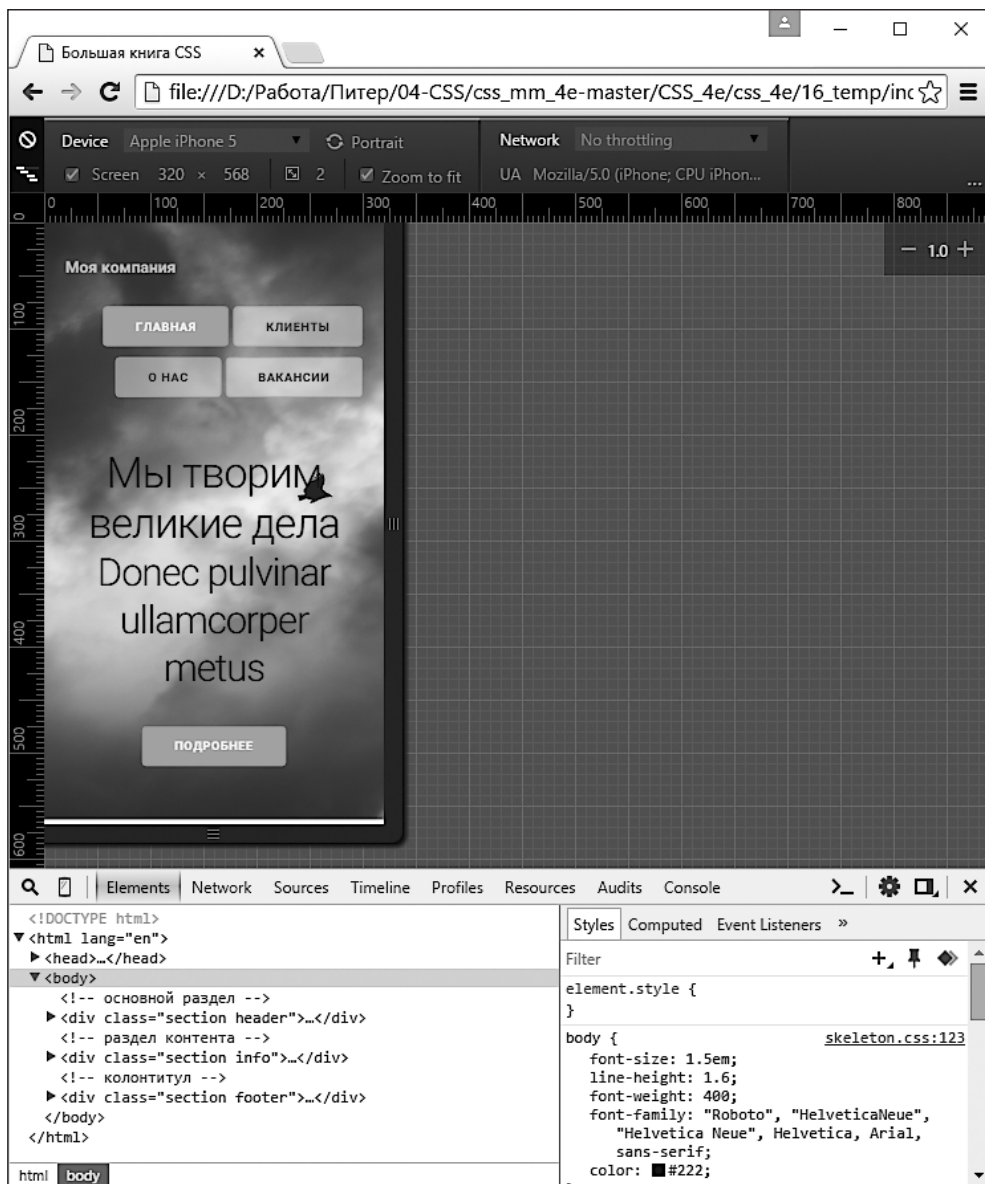


Рис. 16.10. С помощью панели Developer Tools (Инструменты разработчика) браузера Chrome вы можете эмулировать экран различных устройств, включая iPhone 5 (в данном случае)

3. Добавьте следующий стиль под правилом `.nav` в файле `custom.css`:

```
.nav .button {
  width: 48%;
  margin: 2px 0;
}
```


Этот стиль выравнивает ширину всех кнопок на панели навигации, а также настраивает поля, чтобы кнопки не касались друг друга (см. рис. 16.11, *вверху справа*).

В верхней части страницы все еще слишком много пустого пространства.

4. Найдите первый стиль в файле `custom.css` — правило `.header` — и измените значение свойства `padding` на `30px 0 20px 0`; чтобы удалить пустое пространство в верхней части страницы.

Теперь панель навигации и логотип лучше подходят для экранов с небольшим разрешением. Тем не менее под кнопкой **Подробнее** слишком много воздуха (см. рис. 16.11, *внизу слева*). Наконец, было бы неплохо немного уменьшить слишком большой заголовок и добавить пространство под кнопками навигации.

5. Найдите правило `.action` и измените значение свойства `padding-top` на `37px`.

Этот код добавляет некоторое пространство между первым заголовком и кнопками навигации. Теперь необходимо уменьшить размер шрифта заголовков.

6. В файле `custom.css` сразу после правила `.action h2` добавьте два следующих стиля:

```
h1 {  
    font-size: 3rem;  
}  
h2 {  
    font-size: 2.5rem;  
}
```

Если вы сохраните файл и просмотрите его с помощью панели **Developer Tools** (Инструменты разработчика) браузера Chrome и в режиме эмуляции смартфона на iPhone 5, то увидите нечто похожее на показанное на рис. 16.11.

Форматирование точек останова

До этого момента вы занимались базовым дизайном для мобильных устройств. Созданная вами страница выглядит прекрасно на экранах, разрешение по горизонтали которых не превышает 550 пикселей. Другими словами, вам не нужно добавлять какие-либо стили к первому медиазапросу, значение свойства `min-width` которого равно 400 пикселям. Тем не менее при экранном разрешении по горизонтали 550 пикселей и выше дизайн страницы начинает выглядеть немного странно. Это точка, в которой встроенная в систему **Skeleton** модульная сетка изменяется, а дизайн страницы становится многоколоночным.

Как вы можете увидеть на рис. 16.12, кнопки навигации занимают много места и не выровнены. Самое время улучшить внешний вид вашей страницы для экранов с более высоким разрешением по горизонтали.

1. Откройте файл `custom.css` в редакторе HTML-кода. Найдите медиазапрос, значение свойства `min-width` которого равно `550px`, и добавьте следующий код (добавленный код выделен полужирным шрифтом):



Рис. 16.11. Создание дизайна, предназначенного в первую очередь для мобильных устройств, требует постоянной итерации — изменения стилей с целью настройки воздуха, размера шрифта и позиционирования элементов для устройств с различным экранным разрешением

```
/* стили для ширины 550 пикселей и выше */
@media (min-width: 550px) {
  .header {
    padding: 40px 0 50px 0;
  }
  .logo {
    position: static;
    background-color: transparent;
    font-size: 2rem;
    line-height: 1;
  }
  .nav .button {
    width: auto;
  }
  h1 {
    font-size: 5rem;
  }
  h2 {
    font-size: 4.2rem;
  }
}
```

Эти стили сбрасывают пространство в области заголовка страницы, изменяют логотип (он больше не фиксируется на экране), настраивают ширину кнопок и увеличивают размер шрифта заголовков (рис. 16.13, *вверху справа*).

Наконец, вам необходимо увеличить размер шрифта текста логотипа.

2. В медиазапрос, значение свойства `min-width` которого равно 750px, добавьте следующий стиль:

```
/* стили для ширины 750 пикселей и выше */
@media (min-width: 750px) {
  .logo {
    font-size: 3rem;
    position: relative;
    top: 5px;
  }
}
```

Этот код изменяет размер шрифта, который используется для названия компании, и немного смещает его.

3. Сохраните файл `custom.css`. Откройте файл `index.html` в браузере и уменьшите ширину окна таким образом, чтобы она не превышала 550 пикселей. Затем медленно увеличьте ее.

Дизайн страницы, когда окно браузера уже 550 пикселей, показан вверху слева на рис. 16.13. По мере увеличения ширины окна браузера страница будет изменяться два раза: один раз при ширине 550 пикселей (*вверху справа*) и еще раз при ширине 750 пикселей (*внизу*).



Рис. 16.12. Модульная сетка системы Skeleton меняется при отображении на устройствах с экранным разрешением по горизонтали выше 550 пикселей. Медиазапрос начинает действовать, а стили, расположенные в нем, преобразуют страницу, добавляя колонки, заливая в них ваш контент

Вы можете и дальше улучшать дизайн страницы, используя различные точки останова. Попробуйте добавлять разные стили для различных точек останова медиазапросов и посмотреть, как еще можно улучшить этот проект.

Полную версию файлов этого практикума можно найти в папке `16_finished`.

Как вы можете видеть, система модульной верстки представляет собой очень удобный инструмент, который позволяет легко создавать колонки с согласованной шириной. Тем не менее, если присмотреться, вы обнаружите те же принципы, о которых узнали ранее в этом разделе книги: обтекаемые элементы, значения ширины в процентах и т. д. Если вы потратите некоторое время и изучите файл `skeleton.css`, то узнаете о принципах его работы.

СОВЕТ

Статью о том, как создать собственную систему модульной верстки, вы найдете по адресу tinyurl.com/otqef3v.



Рис. 16.13. Система модульной верстки Skeleton позволяет легко создавать веб-страницы, адаптируемые под браузеры различных устройств, начиная со смартфонов и заканчивая телевизорами