



ГЛАВА 3

Макетирование HTML-страниц с фреймворком Bootstrap

Фреймворк Bootstrap позволяет ускорить процесс верстки HTML-страниц и сделать готовый продукт адаптивным к разным вариантам разрешения и формата экрана. Он распространяется бесплатно в виде файлов формата HTML, CSS и JS. Это один из самых популярных фреймворков в мире. Основная область применения — разработка интерфейса сайтов, веб-приложений и среды администрирования. Любому веб-разработчику необходимо знать хотя бы базовые сведения об этом фреймворке и использовать его в своей работе.

Из материалов данной главы вы узнаете:

- о технологиях, заложенных в основу Bootstrap;
- как можно получить файлы фреймворка Bootstrap;
- что такое контейнеры и сетка Bootstrap;
- как можно сверстать макет HTML-страниц;
- как подключить файлы фреймворка Bootstrap к проекту;
- как задать цвет элементам HTML-страниц;
- как задать отступы в элементах макета HTML-страниц;
- как выровнять содержимое в адаптивных блоках HTML-страниц;
- как можно обозначить границы элементов в макете HTML-страниц;
- как использовать адаптивные контейнеры при макетировании HTML-страниц.

3.1. Технологические возможности фреймворка Bootstrap

Bootstrap — это свободный набор инструментов для создания сайтов и веб-приложений. Он включает в себя CSS-шаблоны и JavaScript, которые можно использовать для оформления кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса. Фреймворк Bootstrap применяют по всему миру не только независимые разработчики, но и целые компании.

Современные веб-приложения адаптированы под различные экраны: вся информация должна помещаться в поле экрана без горизонтальной полосы прокрутки. Bootstrap по-

зволяет создать привлекательный дизайн интерфейса приложения и абсолютную адаптивность под разрешения различных экранов.

Bootstrap очень популярен по всему миру, т. к. позволяет верстать сайты в несколько раз быстрее, чем это можно выполнить на «чистом» CSS и JavaScript. Кроме того, популярность этого фреймворка обусловлена доступностью и легкостью использования. Его можно свободно подключить к своему приложению, и даже начинающий разработчик сможет верстать достаточно качественные макеты сайтов, которые трудно было бы выполнить без глубоких знаний веб-технологий и практических навыков.

Bootstrap был создан в Twitter в середине 2010 года и изначально предназначался исключительно для внутренних нужд компании. Однако уже через год состоялся публичный выпуск Bootstrap как фреймворка с открытым исходным кодом. С тех пор были выпущены более двадцати релизов этого фреймворка. Уже в версии Bootstrap 2 была реализована адаптивная функциональность, а в версии Bootstrap 3 библиотека была адаптирована к мобильным устройствам. В Bootstrap 4 проект был серьезно модифицирован, чтобы учесть два ключевых архитектурных изменения: переход на новые таблицы стилей — Sass (Syntactically Awesome Stylesheets) и на гибкие макеты страниц — CSS Flexbox.

Sass — это скриптовый метаязык (т. е. язык, описывающий другой язык), разработанный для упрощения файлов CSS. Синтаксис Sass полностью совместим с синтаксисом CSS, но при этом он короче, в нем отсутствуют скобки и точки с запятой, поэтому набирать код проще. Отступы имеют логическое значение, поэтому крайне важно следить за ними, т. к. неправильный отступ может сломать таблицу стилей. Такой синтаксис определенно понравится тем, кто программирует на Python. Sass дает возможность использовать переменные, вложенные правила, миксины, инлайновые импорты и многое другое, что совместимо с синтаксисом CSS. Sass упрощает создание больших таблиц стилей, а небольшим таблицам стилей позволяет работать быстро.

CSS Flexbox — это технология для создания сложных гибких макетов за счет правильного размещения элементов на странице. С помощью данной технологии можно очень просто и гибко расставить элементы в контейнере, распределить доступное пространство между ними, и выровнять их тем или иным способом, даже если они не имеют конкретных размеров. Технология CSS Flexbox позволяет упростить создание адаптивного дизайна, методы позиционирования пригодны как для разметки целой страницы, так и ее отдельных блоков.

Последний выпуск Bootstrap 5 ориентирован на улучшение кодовой базы Bootstrap 4 с минимальным количеством серьезных критических изменений. Были улучшены существующие функции и компоненты, удалена поддержка старых браузеров и внедрены более перспективные технологии, такие как настраиваемые свойства CSS.

Применение Bootstrap имеет следующие преимущества:

1. Поддерживается всеми популярными браузерами.
2. Он легок в освоении. С помощью знаний основ HTML и CSS каждый начинающий разработчик может создавать качественные веб-приложения. Кроме того, на официальном сайте Bootstrap выложена хорошо структурированная документация.
3. Создает адаптивный дизайн. Приложения, созданные на Bootstrap, сами приспособливаются к десктопам, ноутбукам, планшетам и мобильным телефонам.

4. Он содержит красивые и функциональные встроенные компоненты, которые легко настраивать.
5. Это фреймворк с открытым исходным кодом и находится в свободном доступе.

Bootstrap состоит из следующих элементов:

- инструменты для создания макета приложения (оберточных контейнеров, мощной системы сеток, гибких медиаобъектов, адаптивных утилитных классов);
- классов для стилизации базового контента: текста, изображений и таблиц;
- готовых компонентов: кнопок, форм, горизонтальных и вертикальных навигационных панелей, слайдеров, выпадающих списков, аккордеонов, модальных окон, всплывающих подсказок и др.;
- утилитных классов для решения традиционных задач, наиболее часто возникающих перед веб-разработчиками: выравнивание текста, отображение и скрытие элементов, задание цвета, фона, отступов и т. д.

На текущий момент есть три версии, которые можно использовать в своих проектах:

- v3 — это 3.4.1;
- v4 — это 4.6.0;
- v5 — это 5.0.0.

Bootstrap 5 рекомендуется для проектов, которые предназначены только для современных браузеров (поддержка IE и других браузеров не нужна). В других случаях лучше подключать файлы Bootstrap 4. Третью версию в основном имеет смысл использовать, если нужна поддержка «старых» браузеров (IE8 и IE9).

Материалы данной книги опираются на последнюю версию фреймворка Bootstrap 5. При этом мы не будем разбирать все тонкости и элементы этого фреймворка, а остановимся только на ключевых моментах, касающихся разметки HTML-страниц, т. к. это нам пригодится при формировании шаблонов Django.

3.2. Получение файлов фреймворка Bootstrap

Есть несколько способов использовать фреймворк Bootstrap в своих приложениях:

- подключить свое приложение (сайт) к коду фреймворка через сеть доставки контента CDN (Content Delivery Networks);
- скачать готовые (скомпилированные) файлы и подключить их к своему приложению;
- скачать исходные файлы и скомпилировать (собрать) свой собственный пакет, который будет содержать только необходимые компоненты CSS и плагины JS.

Мы рассмотрим первые два варианта как наиболее простые и доступные.

Использование сети CDN. Наиболее простой способ — подключение приложения (сайта) к фреймворку через сеть доставки контента (CDN). CDN — это географически распределенная сетевая инфраструктура, обеспечивающая быструю доставку контента пользователям веб-сервисов и сайтов. Входящие в состав CDN серверы географически располагаются таким образом, чтобы сделать время ответа для пользователей сайта

(сервиса) минимальным. Для такого подключения Bootstrap страницы сайта в заголовке должны содержать ссылку на CSS- и JS-файлы. Далее приведена HTML-страница, которая содержит эти ссылки:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Bootstrap 5</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link ref="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/
        dist/css/bootstrap.min.css" rel="stylesheet">
    <script>
        src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/
            dist/js/bootstrap.bundle.min.js">
    </script>
</head>
<body>

</body>
</html>
```

Здесь в теге `<head>` заголовка сайта присутствует пять строчек. Первые две строки не имеют отношения в Bootstrap, но для начинающих изучать основы веб-программирования поясним их назначение. Тег `<title>` служит для вывода информации, которая появится на самой верхней вкладке интернет-браузера. Далее следуют две строки с метатегами, разберемся, что это за теги `<meta>` и для чего они нужны.

Тег `<meta>` (от англ. *meta information* — метаинформация) определяет данные, которые предназначены для браузеров и поисковых систем. Например, механизмы поисковых систем обращаются к метатегам для получения описания сайта, ключевых слов и других данных. Разрешается использовать более чем один метатег, все они размещаются в заголовке HTML-страницы — в теге `<head>`.

Метатег `<meta charset="UTF-8">` определяет кодировку символов, которая будет принята на сайте. В приведенном ранее коде тег означает, что на сайте будет использоваться 8-битовый формат преобразования Unicode, который является стандартной кодировкой символов в последней версии HTML 5. Эта строка должна быть включена в каждую созданную веб-страницу, т. к. она гарантирует правильное отображение символов любого языка в любом браузере. Кодировка UTF-8 гарантирует, что символы нелатинских языков не будутискажаться. Браузер Google Chrome автоматически установил кодировку UTF-8, поэтому вам не придется беспокоиться об этом при разработке дизайна для этого браузера. Но вам все равно нужно включать `<meta charset="UTF-8">` в каждый файл HTML на случай, если эта функция не поддерживается другими браузерами.

Метатег `<name="viewport">` определяет, как нужно настроить параметры окна браузера для конкретного устройства, особенно это актуально для мобильного телефона. В настоящее время важно, чтобы веб-страницы хорошо отображались на всех устройствах: на настольных компьютерах, ноутбуках, планшетах и особенно на мобильных телефонах. Для этого нужно включить метатег `<name="viewport">` в каждый файл HTML. Этот тег определяет, как страницы сайта будут отображаться на экранах разного размера и

сколько визуальной области браузера будет доступно пользователю. Каждое устройство имеет разные размеры и разрешение экрана. Как правило, у мобильных устройств видимая область экрана меньше, а у настольных компьютеров больше. Использование параметра `content="width=device-width"` — первый шаг к тому, чтобы страницы сайтов хорошо выглядели на мобильных устройствах. Это не позволяет страницам сайта, который просматривается на мобильном устройстве, выглядеть так, как если бы он отображался на ноутбуке, т. е. сильно уменьшенным с мелкими и плохо читаемыми символами. Это гарантирует, что HTML-страницы будут автоматически адаптированы под размеры экрана устройства. Параметр `initial-scale=1.0` устанавливает начальный масштаб при первой загрузке страницы браузером, т. е. разворачивает его на полный экран.

В дополнение к приведенным метатегам на странице можно разместить еще три полезных метатега.

Метатег `<x-ua-compatible>` отвечает за браузеры Microsoft IE8, IE9, IE10, IE11. Он говорит о том, что страница будет адаптирована к последней версии Internet Explorer. Всегда выбирайте последнюю версию Internet Explorer, т. е. `IE=edge`. Вот пример строки с данным метатегом:

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

Метатег `<meta name="description">`. Наличие этого метатега помогает поисковым системам ранжировать ваш сайт по сравнению с другими сайтами. Он предназначен в основном для целей SEO (поисковая оптимизация). Метатег описания позволяет кратко и лаконично объяснить, о чем ваш сайт, и может выглядеть примерно так:

```
<meta name="description" content="Разработка веб-приложений: Python, Django, Bootstrap, HTML. Программирование на Python.">
```

Что нужно учитывать при описании вашего сайта в этом метатеге:

- описание должно быть коротким, не превышающим 160 символов;
- в описании должны быть ключевые слова, по которым люди ищут услуги, предоставляемые вашим сайтом;
- четко укажите ваши преимущества перед конкурентами.

Метатег `<meta name="author">`. Еще один полезный метатег, который следует включить в описание сайта, — это имя автора:

```
<meta name="author" content="Семенов Алексей">
```

Эта информация показывает, кто является владельцем веб-сайта или кому принадлежат авторские права.

Скачивание файлов Bootstrap. Второй путь использования фреймворка Bootstrap — скачать готовые файлы и подключить их к своему приложению. Этот путь тоже не представляет особых трудностей. Скачать готовые файлы можно с официального сайта по следующей ссылке: <https://getbootstrap.com/docs/5.2/getting-started/download/>.

После скачивания вы получите архивированный zip-файл, в имени которого будет отражена текущая версия фреймворка, например `bootstrap-5.2.3-dist.zip`. После распаковки архива будут созданы две папки: `css` — с файлами стилей и `js` — с java-скриптами. Эти две папки нужно перенести в ваш проект.

Подключение файлов Bootstrap. Последние две строки обеспечивают подключение приложения к файлам CSS фреймворка Bootstrap через сеть доставки контента CDN:

```
<link href="https://cdn... — подключение приложения к файлам CSS;
```

```
<script src="https://cdn... — подключение приложения к файлам JS.
```

Вот, собственно, и все, в таком виде на HTML-странице можно использовать все доступные в Bootstrap классы, методы и свойства.

3.3. Контейнеры и сетка Bootstrap

Сначала разберемся с тем, что такое адаптивный макет сайта. Адаптивный макет — это такой макет, вид которого может изменяться в зависимости от того, какую ширину (viewport) имеет браузер. Это означает, что при одних значениях ширины viewport адаптивный макет может выглядеть одним образом, а при иных — совершенно по-другому. В Bootstrap изменение вида макета реализовано через медиазапросы. Каждый медиазапрос в Bootstrap строится на основании минимальной ширины браузера. В Bootstrap ключевое значение ширины viewport в медиазапросе называется *breakpoint* (контрольной точкой, или классом устройства). На рис. 3.1 приведены основные контрольные точки, которые в Bootstrap 4 и Bootstrap 5 назначены по умолчанию.

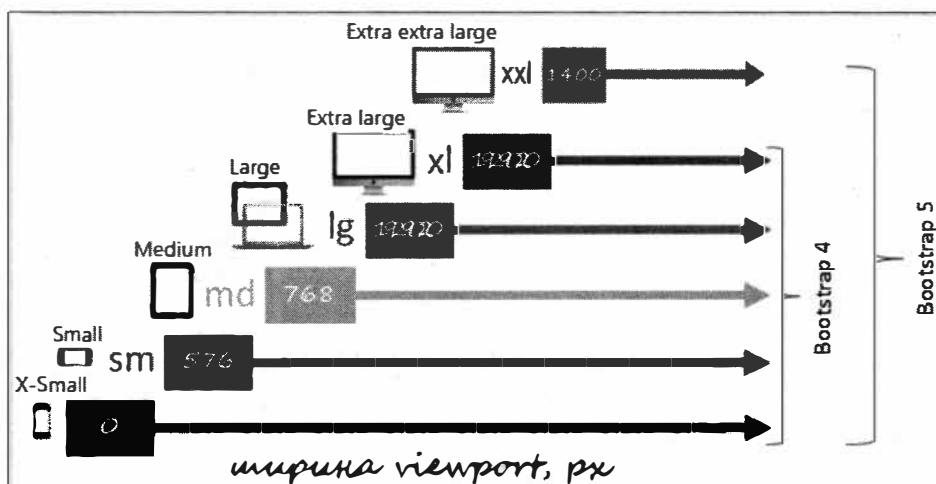


Рис. 3.1. Контрольные точки для Bootstrap 4 и Bootstrap 5

Цифры на этом рисунке означают, что при ширине видимой области браузера до 576 px макет сайта будет отображаться одним образом, от 576 до 768 px — другим образом и т. д. Иными словами, можно создать макет, который на каждом из этих участков видимой области браузера станет выглядеть по-разному.

Контрольные точки (breakpoint) имеют обозначение. Первая контрольная точка не имеет обозначения (т. е. `xs` не указывается), вторая обозначается — `sm`, третья — `md`, четвертая — `lg`, пятая — `xl` и шестая — `xxl`. Эти обозначения необходимо запомнить, т. к. они используются в классах, которые мы будем добавлять к элементам. Эти обозначения в имени класса будут указывать на то, с какой ширины `viewport` стили, определенные

в нем, будут применяться к вложенным элементам. При этом контрольные точки задают только минимальную ширину. Например, если вы определили макет, указав в нем классы без обозначения контрольной точки (`xs`) и с использованием параметра `md`, то на устройстве `sm` страница будет иметь такую же структуру, как и на устройстве `xs`, а на устройствах с большим экраном `lg`, `xl` и `x1l` — как на `md`.

Теперь разберемся с сеткой HTML-страницы в Bootstrap. Сетка состоит из следующих элементов:

- оберточных контейнеров (элементов с классом `container`, `container-fluid`, или `container-{breakpoint}`);
- рядов или строк (элементов с классом `row`);
- адаптивных колонок или блоков, имеющих один или несколько классов `col`.

Все части сетки — это обычные элементы HTML, к которым просто добавлены определенные классы.

3.3.1. Адаптивные контейнеры

Адаптивный (или оберточный) контейнер — это элемент сетки Bootstrap, с которого обычно начинается создание макета страницы или ее части. Другими словами, это базовый элемент, в котором необходимо размещать все другие элементы сетки (ряды и адаптивные блоки). Его основная цель — это установить шаблону ширину и выровнять его по центру страницы (рис. 3.2).



Рис. 3.2. Оберточный контейнер

Контейнеры в Bootstrap являются основным элементом макетирования страниц. Контейнеры служат для вкладывания в них других элементов интерфейса, формирования отступов и выравнивания или центрирования их содержимого. Хотя контейнеры могут быть вложенными, для большинства макетов вложенный контейнер не требуется.

Bootstrap поставляется с двумя базовыми контейнерами:

- адаптивно фиксированный контейнер (класс `container`) — устанавливает максимальную ширину (`max-width`) для каждой контрольной точки;
- адаптивно подвижный («резиновый») контейнер (класс `container-fluid`), который устанавливает максимальную ширину (`width: 100%`) во всех контрольных точках.

Для адаптивно фиксированного контейнера можно указать контрольную точку (`container-{breakpoint}`), при этом будет установлена максимальная ширина (`width: 100%`) до указанной контрольной точки.

Адаптивно-фиксированный контейнер предназначен для создания контейнера с постоянной шириной, которая будет оставаться постоянной только в пределах действия

определенной контрольной точки. Например, на контрольной точке sm до действия контрольной точки md он будет иметь одну фиксированную ширину, а на md до действия lg — другую фиксированную ширину. Единственная контрольная точка, на которой данный контейнер не будет иметь фиксированную ширину, — это breakpoint без обозначения. Здесь контейнер будет занимать 100%-ную ширину видимой области браузера. Пример расположения адаптивно-фиксированного контейнера в окне браузера представлен на рис. 3.3.



Рис. 3.3. Расположения адаптивно фиксированного контейнера в окне браузера

Адаптивно-фиксированный контейнер будет иметь следующие значения ширины:

- 100%-ную ширину при ширине viewport до 576 px;
- 540 px при ширине viewport от 576 до 768 px;
- 720 px при ширине viewport от 768 до 992 px и т. д.

В горизонтальном направлении контейнер располагается по центру окна браузера. Осуществляется это посредством установки ему CSS-свойств margin-left:auto и margin-right:auto в файле bootstrap.css. Создать адаптивно-фиксированный контейнер можно так:

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

Второй вид контейнера — это адаптивно-подвижный, или «резиновый» контейнер. Он применяется тогда, когда необходимо создать полностью гибкий макет целой страницы или ее части. Данный контейнер на любых контрольных точках занимает полную (100%-ную) ширину видимой области браузера (рис. 3.4).



Рис. 3.4. Расположение адаптивно-подвижного контейнера в окне браузера

Создать адаптивно-подвижный контейнер можно так:

```
<div class="container-fluid">...</div>
```

Контейнеры классов `container` и `container-fluid` имеют внутренние отступы слева и справа по 15 px. Установка внутренних отступов контейнеров задается в файле `bootstrap.css` посредством CSS-свойств `padding-left:15px` и `padding-right:15px`. При верстке макета страницы не следует одни оберточные контейнеры помещать внутрь других.

3.3.2. Ряды или строки (row)

Ряд или строка — это элемент сетки страницы, который выступает в роли контейнера для адаптивных блоков (колонок). Это дочерний элемент контейнера и родительский элемент для адаптивных блоков (рис. 3.5).

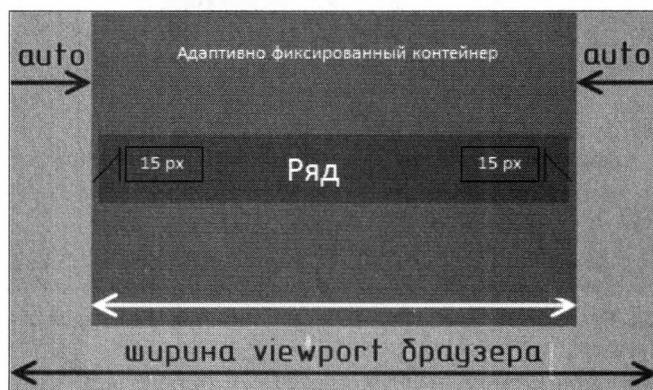


Рис. 3.5. Расположение ряда (строки) в теле адаптивно-фиксированного контейнера

Для того чтобы создать ряд, нужно использовать следующий код:

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

В Bootstrap ряды играют очень важную роль. Это связано с тем, что сетка HTML-страниц построена на CSS Flexbox — адаптивных блоках. В этой сетке ряд выступает в роли flex-контейнера для flex-элементов (адаптивных блоков). Если адаптивные блоки окажутся вне ряда, то они работать не будут, в Bootstrap адаптивные блоки должны обязательно находиться в блоке с классом `row`. Компенсация внутренних отступов (padding) осуществляется за счет отрицательных левых и правых внешних отступов, равных 15 px (`margin-left:-15px` и `margin-right:-15px`).

3.3.3. Адаптивные блоки (col)

Адаптивные блоки — это основные строительные элементы сетки. Именно от них зависит то, как макет веб-страницы будет отображаться в видимой области браузера на разных контрольных точках. Адаптивные блоки располагаются внутри ряда, т. е. ряд является контейнером для адаптивных блоков (рис. 3.6).

Адаптивные блоки — это блоки, ширина которых на разных контрольных точках (breakpoint) может быть различной (в процентном отношении от родительского кон-

тейнера). Например, адаптивный блок в браузере смартфона (`sm`) может иметь ширину, равную 50% от родительского элемента, а на планшете (`md`) — 25%.

Адаптивный блок достаточно просто создать посредством добавления одного или нескольких классов `col-?-?` к HTML-элементу, расположенному в ряду. В классе `col-?-?` вместо первого знака вопроса указывается название контрольной точки («пусто», `sm`, `md`, `lg` или `xxl`). Вместо второго знака вопроса задается ширина адаптивного блока, которую он должен иметь на указанной контрольной точке. Ширина адаптивного блока назначается в относительной форме либо по умолчанию, либо с помощью числа от 1 до 12 (по количеству колонок).

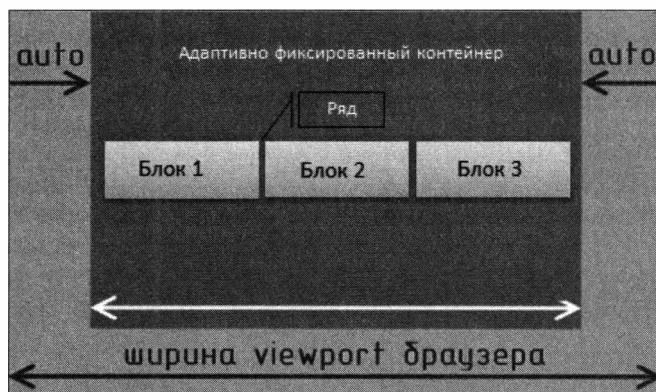


Рис. 3.6. Расположение адаптивных блоков в контейнере в ряд (в строке)

Число, которое стоит при инициализации класса `col`, определяет, какую часть ширины родительского контейнера (т. е. ряда) будет занимать адаптивный блок, начиная с указанной контрольной точки. При этом ширина ряда в числовом выражении (в колонках Bootstrap) по умолчанию равна 12. Например, блок с классом `col-md-4`, начиная с контрольной точки `md`, будет занимать $4/12 \cdot 100\% = 33,3\%$.

Адаптивные блоки, как и оберточные контейнеры, имеют внутренние отступы слева и справа по 15 px. Данные отступы адаптивным блокам во фреймворке Bootstrap устанавливаются с помощью CSS свойств `padding-left:15px` и `padding-right:15px`. Размещать адаптивные блоки необходимо в ряду (в строке), т. е. у любого адаптивного блока в качестве родителя должен быть обязательно элемент `row`.

Рассмотрим на примерах, какую ширину будут иметь следующие адаптивные блоки.

1. `class="col-12"` задает ширину блока по умолчанию. Ширина блока будет равна 12 колонкам Bootstrap (т. е. $12/12 \cdot 100\% = 100\%$ от ширины ряда), эту ширину блок будет иметь, начиная с контрольной точки `xs`.
2. `class="col-sm-9"` задает ширину блока с контрольной точки `sm`. Ширина блока будет равна 9 колонкам Bootstrap (т. е. $9/12 \cdot 100\% = 75\%$ от ширины ряда), эту ширину блок будет иметь, начиная с контрольной точки `sm`.
3. `class="col-md-7"` задает ширину блока с контрольной точки `md`. Ширина блока будет равна 7 колонкам Bootstrap (т. е. $7/12 \cdot 100\% = 58,3\%$ от ширины ряда), начиная с контрольной точки `md`.

4. class="col-1g-5" задает ширину блока с контрольной точки lg. Ширина блока будет равна 5 колонкам Bootstrap (т. е. $5/12 \cdot 100\% = 41,6\%$ от ширины ряда), начиная с контрольной точки lg.
5. class="col-xl-3" задает ширину блока с контрольной точки xl. Ширина блока будет равна 3 колонкам Bootstrap (т. е. $3/12 \cdot 100\% = 25\%$ от ширины ряда), начиная с контрольной точки xl.
6. class="col-xxl-6" задает ширину блока с контрольной точки больше xxl. Ширина блока будет равна 2 колонкам Bootstrap (т. е. $6/12 \cdot 100\% = 50\%$ от ширины ряда).

При задании ширины адаптивному блоку мы указываем класс, содержащий контрольную точку, начиная с которой данная ширина будет действовать. Эту ширину данный блок будет иметь до тех пор, пока она не будет переопределена с помощью другого класса, действие которого начинается с наибольшей ширины viewport.

3.3.4. Адаптивные блоки без указания числа колонок

В сетке Bootstrap имеются специальные классы col, col-sm, col-md, col-lg, col-xl, col-xxl, col-auto, col-sm-auto, col-md-auto, col-lg-auto, col-xl-auto и col-xxl-auto.

Первая группа классов (col, col-sm, col-md, col-lg, col-xl, col-xxl) предназначена для создания адаптивных блоков, ширина которых будет зависеть от свободного пространства в ряду (строке). Незанятая ширина (свободное пространство) в ряду между всеми такими блоками распределяется равномерно. Кроме того, данные адаптивные блоки перед распределением свободного пространства ряда (по умолчанию) имеют нулевую ширину.

3.3.5. Расположение адаптивных блоков

Адаптивные блоки по умолчанию располагаются в ряду горизонтальными линиями, при этом они выстраиваются последовательно слева направо. В одну горизонтальную линию могут поместиться адаптивные блоки с суммарным числом колонок не более 12.

Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8	Col 9	Col 10	Col 11	Col 12						
Col 6						Col 6											
Col 4				Col 4				Col 4									
Col 3		Col 3			Col 3			Col 3		Col 3							
Col 2		Col 2		Col 2		Col 2		Col 2		Col 2							
Col 8						Col 4											
Col 10										Col 2							
Col 4			Col 8														
Col 2		Col 10															

Рис. 3.7. Возможные варианты формирования адаптивных блоков из 12 колонок

Адаптивные блоки, которые не помещаются в текущую строку, переходят на следующую. Колонки можно объединять в адаптивные блоки произвольным образом. Пример различных вариантов расположения колонок в адаптивных блоках в ряду приведен на рис. 3.7.

3.4. Верстка макета HTML-страниц

Основной принцип верстки макета веб-страницы на сетке Bootstrap заключается во вкладывании адаптивных блоков в ряды, а рядов — в контейнеры. При этом ширина адаптивных блоков — это всегда относительная величина, которая задается в колонках Bootstrap и зависит только от ширины родителя, т. е. ряда. Размещать контент веб-страницы следует только в адаптивных блоках. Вот пример формирования макета страницы:

```
<div class="container">
  <div class="row">
    <div class=col-8>Блок 1</div>
    <div class=col-4>Блок 2</div>
  </div>
</div>
```

В этом примере был создан контейнер (`class="container"`), в него вложена строка (`class="row"`), и уже в эту строку вложено два адаптивных блока: шириной 8 колонок (`class=col-8`) и шириной 4 колонки (`class=col-4`).

Если предполагается, что блоки будут одинаковой ширины, то число колонок для них можно не указывать, например:

```
<div class="container">
  <div class="row">
    <div class="col">Блок 1 </div>
    <div class="col">Блок 2 </div>
    <div class="col">Блок 3 </div>
    <div class="col">Блок 4 </div>
    <div class="col">Блок 5 </div>
  </div>
</div>
```

Выравнивание адаптивных блоков в горизонтальном и вертикальном направлениях осуществляется с помощью служебных flex-классов.

В пределах ряда по горизонтали адаптивные блоки выравнивают посредством одного из следующих классов, который необходимо дополнительно добавить к `row`:

- `align-items-start` — выравнивание относительно начала ряда (по левому краю);
- `align-items-center` — по центру ряда;
- `align-items-end` — относительно конца ряда (по правому краю).

Например, в следующем коде адаптивные блоки выровнены по центру ряда:

```
<div class="row align-items-center">
  <div class="col">Блок 1</div>
  <div class="col">Блок 2 </div>
</div>
```

По умолчанию адаптивные блоки занимают всю высоту ряда, в котором они расположены. Выравнивание какого-то определенного адаптивного блока по вертикали в пределах ряда может осуществляться одним из следующих классов:

- align-self-start — выравнивание относительно начала ряда (по верху ряда);
- align-self-center — по центру ряда;
- align-self-end — относительно конца ряда (по низу ряда).

Данные классы необходимо добавлять к адаптивным блокам, а не к ряду. Например, выравниваем адаптивный блок 2 по нижнему краю ряда:

```
<div class="row align-items-center">
    <div class="col">Блок 1</div>
    <div class="col align-self-end">Блок 2</div>
</div>
```

Горизонтальное выравнивание адаптивных блоков. Для выравнивания адаптивных блоков в горизонтальном направлении можно также использовать следующие классы:

- justify-content-start — относительно начала ряда (по левому краю), установлено по умолчанию;
- justify-content-center — по центру ряда;
- justify-content-end — относительно конца ряда (по правому краю);
- justify-content-around — равномерно, с учетом пространства перед первым и последним адаптивными блоками;
- justify-content-between — равномерно, с одинаковым пространством между адаптивными блоками.

Например, распределим адаптивные блоки в горизонтальном направлении равномерно:

```
<div class="row justify-content-around">
    <div class="col-4">Блок 1</div>
    <div class="col-4">Блок 2</div>
</div>
```

Смещение адаптивных блоков в Bootstrap можно выполнить с помощью следующих классов:

- классов offset — на определенное число колонок;
- служебных (утилитных) классов margin.

Классы offset предназначены для смещения адаптивных блоков вправо на определенное число колонок. Данные классы имеют следующий синтаксис:

- offset-{1};
- offset-{breakpoint}-{1}.

Здесь {breakpoint} — контрольная точка, начиная с которой к данному блоку будет применено смещение. Если она не указана, то смещение будет применено, начиная с самых крохотных устройств (смартфоны). Второй параметр {1} — величина смещения (число колонок). Рассмотрим следующий код:

```
<div class="row">
    <div class="col-4">Блок 1</div>
    <div class="col-4 offset-4">Блок 1</div>
</div>
<div class="row">
    <div class="col-3 offset-3">Блок 3 </div>
    <div class="col-3 offset-3">Блок 4</div>
</div>
<div class="row">
    <div class="col-6 offset-3">Блок 5</div>
</div>
```

В этом случае смещение адаптивных блоков будет выглядеть так, как показано на рис. 3.8.

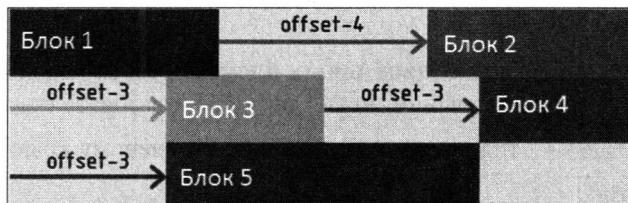


Рис. 3.8. Пример смещения адаптивных блоков с использованием класса offset

Смещение с помощью классов margin. В Bootstrap смещение адаптивным блокам также можно устанавливать с помощью отступов margin (margin-left:auto и margin-right auto). Этот вариант смещения появился благодаря тому, что сетка в новых версиях Bootstrap основывается на CSS Flexbox. Данный вариант удобен, когда блоки необходимо сместить относительно друг друга на некоторую переменную величину. Для более удобного задания блокам отступов margin можно указать сокращенное обозначение классов: ml-auto, mr-auto, ml-(breakpoint)-auto и mr-(breakpoint)-auto. Рассмотрим следующий код:

```
<div class="row">
    <div class="col-4">Блок 1</div>
    <div class="col-4 ml-auto">Блок 2</div>
</div>
<div class="row">
    <div class="col-3">Блок 3</div>
    <div class="col-3 ml-auto mr-auto">Блок 4</div>
    <div class="col-3">Блок 5</div>
</div>
<div class="row">
    <div class="col-4 ml-auto mr-auto">Блок 6</div>
    <div class="col-4 ml-auto mr-auto">Блок 7</div>
</div>
```

В этом случае смещение адаптивных блоков будет выглядеть так, как показано на рис. 3.9.

Итак, мы рассмотрели основные «строительные блоки», из которых создаются макеты HTML-страниц. Адаптивные блоки являются конечными элементами макета, в которые и размещается основной контент HTML-страниц.

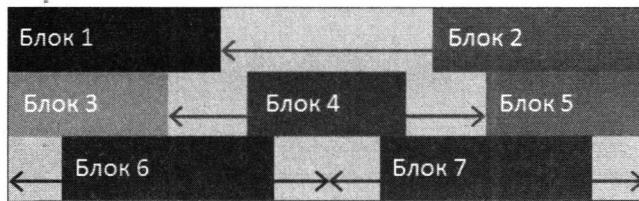


Рис. 3.9. Пример смещения адаптивных блоков с использованием класса margin

3.5. Подключение файлов фреймворка Bootstrap к проекту

Для изучения возможностей фреймворка Bootstrap откроем PyCharm и создадим в нем проект с именем Boot_Start. В этом проекте создадим папку static и в нее перенесем две папки со скачанными файлами Bootstrap: css и js. Структура папок проекта приведена на рис. 3.10.

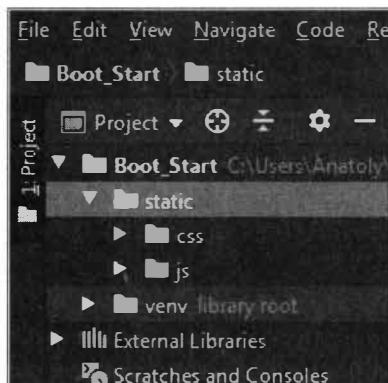


Рис. 3.10. Структура папок проекта Boot_Start

Чтобы процесс создания макетов страниц был более понятен, разберем несколько простых примеров. Для начала создадим контейнер фиксированной ширины с одним рядом (строкой), в котором находится один адаптивный блок из 12 колонок, т. е. он будет занимать всю ширину ряда (листинг 3.1, страница start1.html).

Листинг 3.1. Страница start1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>start1</title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="static/css/bootstrap.min.css" >
  <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
```

```
<body>
<div class="container">
    <div class="row">
        <div class="col">
            <h1>Это страница Bootstrap</h1>
            <p>На странице создан контейнер фиксированной ширины
                class="container".</p>
            <p>В контейнер вложена одна строка class="row."</p>
            <p>В строке находится один адаптивный блок class="col".</p>
            <p>С изменением размера окна браузера ширина элементов
                контейнера будет меняться в разных контрольных точках.</p>
        </div>
    </div>
</div>
</body>
</html>
```

При минимальном размере окна браузера данная страница будет выглядеть как на рис. 3.11.

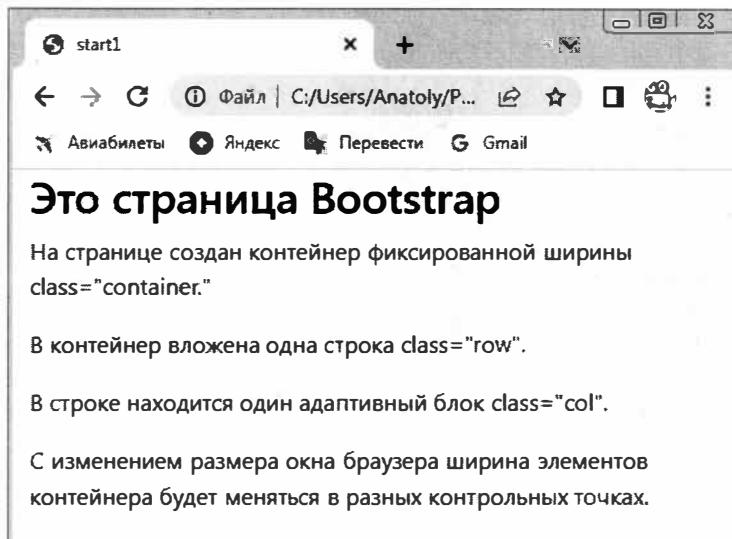


Рис. 3.11. Простейшая HTML-страница, созданная на основе Bootstrap

Как видно из данного рисунка, с параметрами по умолчанию текст на странице показан черным цветом на белом фоне, при этом отсутствуют какие-либо признаки, по которым можно определить размер и положение адаптивного блока, например рамки или выделение блока цветом. Для того чтобы наглядно продемонстрировать возможности макетирования страниц, необходимо явно показать границы блока. Это можно сделать либо с помощью цветного фона, либо созданием ограничительной рамки.

3.6. Задание цвета элементам HTML-страниц

Для задания цвета в Bootstrap предусмотрено несколько полезных классов. Класс `bg` позволяет задать 9 цветов:

- `class="bg-primary";`
- `class="bg-secondary";`
- `class="bg-success";`
- `class="bg-danger";`
- `class="bg-warning";`
- `class="bg-info";`
- `class="bg-light";`
- `class="bg-dark";`
- `class="bg-white".`

Пример HTML-кода страницы, в которой показано использование класса `bg` для задания цвета фону, приведен в листинге 3.2, страница `color_bg.html`.

Листинг 3.2. Страница `color_bg.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>color_bg</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container">
    <div class="row">Первая строка</div>
    <div class="row">
        <div class="col bg-primary">Блок 1</div>
        <div class="col bg-secondary">Блок 2</div>
        <div class="col bg-success">Блок 3</div>
        <div class="col bg-danger">Блок 4</div>
        <div class="col bg-warning">Блок 5</div>
        <div class="col bg-info">Блок 6</div>
        <div class="col bg-light">Блок 7</div>
        <div class="col bg-dark">Блок 8</div>
        <div class="col bg-white">Блок 9</div>
    </div>
</div>
</body>
</html>
```

В окне браузера данная страница будет иметь вид, представленный на рис. 3.12.

Как видно из рис. 3.12, текст «Блок 8» не виден, т. к. и шрифт, и фон имеют одинаковый (черный) цвет.

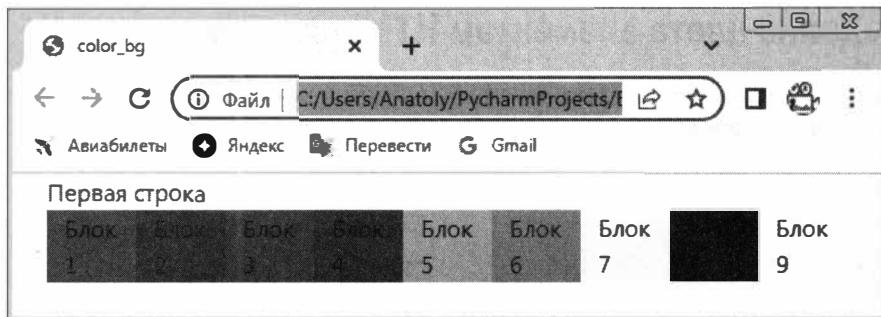


Рис. 3.12. Образцы цвета фона в Bootstrap

Класс `text` позволяет задать 10 цветов:

- `class="text-primary";`
- `class="text-secondary";`
- `class="text-success";`
- `class="text-danger";`
- `class="text-warning";`
- `class="text-info";`
- `class="text-light";`
- `class="text-dark";`
- `class="text-muted";`
- `class="text-white".`

Пример HTML-кода страницы, в которой показано использование класса `text` для задания цвета шрифту, приведен в листинге 3.3, страница `color-text.html`.

Листинг 3.3. Страница `color-text.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>color|_ text</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container">
    <div class="row">Первая строка</div>
    <div class="row">
        <div class="col text-primary">Блок 1</div>
        <div class="col text-secondary">Блок 2</div>
        <div class="col text-success">Блок 3</div>
        <div class="col text-danger">Блок 4</div>
        <div class="col text-warning">Блок 5</div>
        <div class="col text-info">Блок 6</div>
        <div class="col text-light">Блок 7</div>
        <div class="col text-dark">Блок 8</div>
        <div class="col text-muted">Блок 9</div>
        <div class="col text-white">Блок 10</div>
    </div>
</div>
```

```
</div>
</body>
</html>
```

В окне браузера данная страница будет иметь вид, представленный на рис. 3.13.

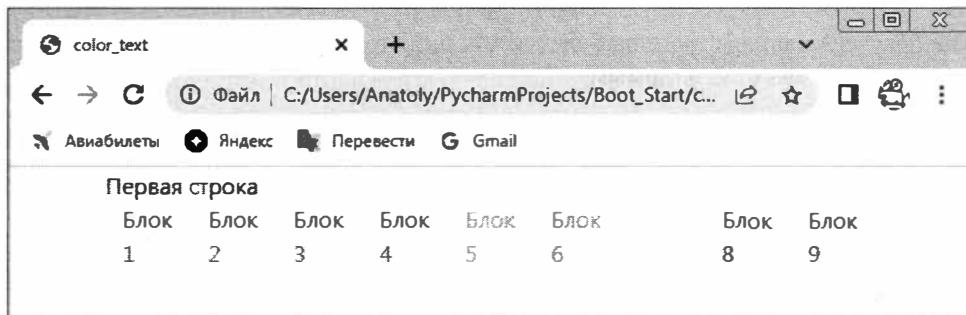


Рис. 3.13. Образцы цвета текста в Bootstrap

Как видно из данного рисунка, текст «Блок 7» и «Блок 10» не виден, т. к. и шрифт, и фон имеют одинаковый цвет.

Для того чтобы текст всегда был виден, он должен иметь цвет, отличный от цвета фона. Пример HTML-кода страницы, в которой показано использование двух классов `text` и `bg` для одновременного задания цвета для шрифта и для фона, приведен в листинге 3.4, страница `color_text_bg.html`.

Листинг 3.4. Страница `color_text_bg.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>color_text_bg</title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="static/css/bootstrap.min.css" >
  <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
  <div class="container">
    <div class="row">Первая строка</div>
    <div class="row">
      <div class="col text-primary bg-info">Блок1</div>
      <div class="col text-secondary">Блок2</div>
      <div class="col text-success bg-info">Блок3</div>
      <div class="col text-danger">Блок4</div>
      <div class="col text-warning bg-info">Блок5</div>
      <div class="col text-info">Блок6</div>
      <div class="col text-light bg-primary">Блок7</div>
      <div class="col text-dark">Блок8</div>
```

```

<div class="col text-muted bg-info">Блок9</div>
<div class="col text-white bg-primary">Блок10</div>
</div>
</div>
</body>
</html>

```

В окне браузера данная страница будет иметь вид, представленный на рис. 3.14. Теперь текст во всех блоках стал видимым.

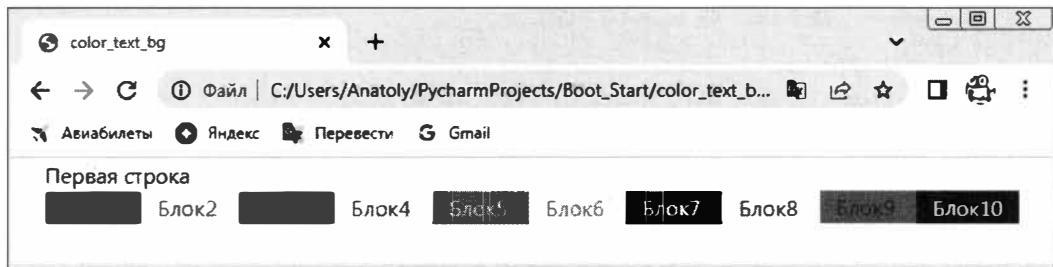


Рис. 3.14. Использование классов `text` и `bg` для задания цвету тексту и фону

3.7. Задание отступов элементам макета HTML-страниц

По умолчанию все адаптивные блоки «прилипают» друг к другу, однако достаточно часто требуется отделить их друг от друга дополнительным пространством. Особенно это актуально в тех случаях, когда они имеют рамки обрамления различных типов. Для задания отступов (разрывов) в Bootstrap существуют два элемента: `margin` и `padding`. Здесь `margin` (имеет сокращение — `m`) и `padding` (имеет сокращение — `p`) — это уникальные свойства CSS, которые добавляют отступы между элементами HTML-страницы и их содержимым. Свойство `margin` добавляет отступы за пределами элемента, а `padding` — внутри элемента (например, контейнера). Синтаксис для задания отступов:

`{property}{sides}-{size}`

Здесь приняты следующие обозначения:

- `property` — свойство;
- `sides` — сторона;
- `size` — размер.

Для свойства (`property`) в коде программ допустимы следующие сокращения:

- `t` — для классов `margin`, которые устанавливают отступы за пределами элемента;
- `p` — для классов `padding`, которые устанавливают отступы внутри элемента.

Параметр `сторона` (`sides`) может принимать сокращенные значения, которые устанавливают следующие отступы:

- `t` — отступ в верхней части элемента (`margin-top` ИЛИ `padding-top`);
- `b` — в нижней части элемента (`margin-bottom` ИЛИ `padding-bottom`);

- s — в стартовой части элемента (start), т. е. слева (margin-left или padding-left);
- e — в конечной части элемента (end), т. е. справа (margin-right или padding-right);
- x — с двух сторон элемента по горизонтали (*-left и *-right);
- y — с двух сторон элемента по вертикали (*-top и *-bottom);
- blank — отступ со всех четырех сторон элемента.

Параметр размер (size) может принимать следующие значения:

- 0 — нет отступов;
- 1 — отступ, равный \$spacer * .25;
- 2 — отступ, равный \$spacer * .5;
- 3 — отступ, равный \$spacer;
- 4 — отступ, равный \$spacer * 1.5;
- 5 — отступ, равный \$spacer * 3;
- auto — отступ, равный авто.

ПРИМЕЧАНИЕ

Здесь \$spacer = 1rem Rem — это единица типографики, равная корневому (базовому) значению размера шрифта (font-size). Это значит, что 1rem всегда будет равен значению font-size, которое было определено в HTML. Для большинства браузеров размер шрифта по умолчанию 16 px до тех пор, пока его не поменяют в настройках браузера (редко кто подобным занимается). Поэтому, как правило, \$spacer = 1rem=16px.

Пример HTML-кода страницы, в которой показано использование отступов для разделения адаптивных блоков, приведен в листинге 3.5, страница space_1.html.

Листинг 3.5. Страница space_1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>space_1</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container">
    <div class="row bg-success text-white">Первая строка</div>
    <div class="row">
        <div class="col mx-2 bg-primary text-white">Блок 1</div>
        <div class="col mx-2 bg-primary text-white">Блок 2</div>
        <div class="col mx-2 bg-primary text-white">Блок 3</div>
    </div>
</div>
</body>
</html>
```

Здесь для адаптивных блоков были вставлены горизонтальные отступы в две позиции (`mx-2`). В окне браузера данная страница будет иметь вид, представленный на рис. 3.15.

Как видно из данного рисунка, между адаптивными блоками появились горизонтальные отступы, однако между рядами (строками) нет разрывов, и они вплотную примыкают друг к другу. Исправим это и вставим отступы между рядами (листинг 3.6, страница `space_2.html`).

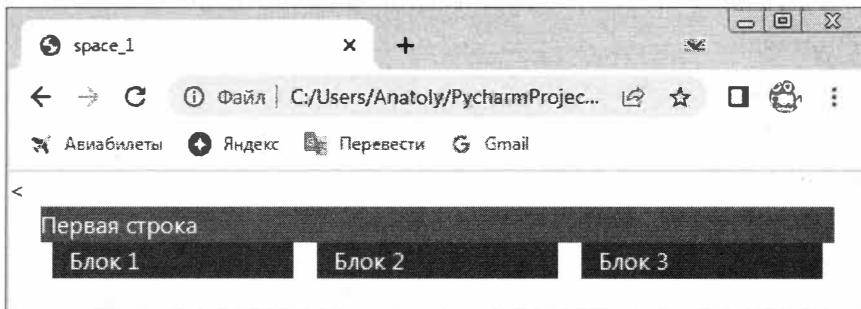


Рис. 3.15. Использование горизонтальных отступов между адаптивными блоками

Листинг 3.6. Страница `space_2.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>space_2</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container">
    <div class="row bg-success text-white">Первая строка</div>
    <div class="row my-2">
        <div class="col mx-2 bg-primary text-white">Блок 1</div>
        <div class="col mx-2 bg-primary text-white">Блок 2</div>
        <div class="col mx-2 bg-primary text-white">Блок 3</div>
    </div>
</div>
</body>
</html>
```

Здесь для второго ряда был вставлен вертикальный отступ в две позиции (`my-2`). В окне браузера эта страница будет иметь вид, представленный на рис. 3.16.

Как видно из данного рисунка, между рядами появился вертикальный отступ.

Теперь посмотрим, как можно создать отступы внутри элемента. Пример HTML-кода страницы, в которой показано использование отступов в теле адаптивных блоков, приведен в листинге 3.7, страница `space_3.html`.

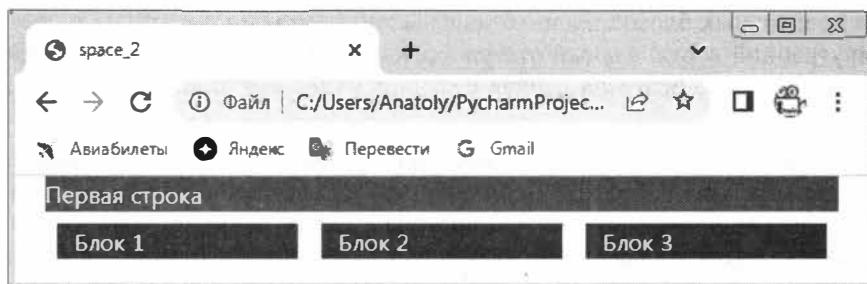


Рис. 3.16. Использование вертикальных отступов между рядами

Листинг 3.7. Страница space_3.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>space_3</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container">
    <div class="row bg-success text-white">Первая строка</div>
    <div class="row my-2">
        <div class="col mx-2 py-5 bg-primary text-white">Блок 1</div>
        <div class="col mx-2 py-5 bg-primary text-white">Блок 2</div>
        <div class="col mx-2 py-5 bg-primary text-white">Блок 3</div>
    </div>
</div>
</body>
</html>
```

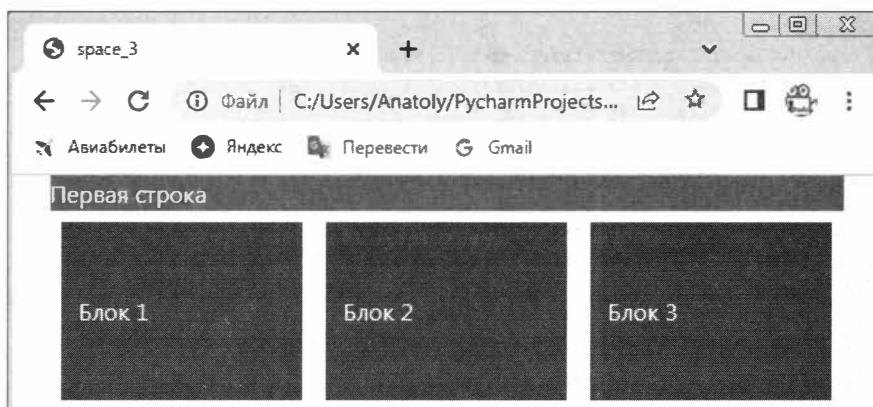


Рис. 3.17. Использование внутренних отступов в теле адаптивных блоков

Здесь для адаптивных блоков были вставлены горизонтальные отступы в две позиции (`mx-2`) и внутренний вертикальный отступ (между текстом и внешней границей) в пять позиций (`py-5`). В окне браузера данная страница будет иметь вид, представленный на рис. 3.17.

Как видно из рис. 3.17, между текстом внутри адаптивных блоков и их внешней границей появились отступы. Обратите внимание, что на всех этих HTML-страницах текст внутри адаптивных блоков по умолчанию прижимается к левому краю, т. е. к началу блока. Разберемся теперь, как можно выравнивать содержимое блоков HTML-страниц.

3.8. Выравнивание содержимого в адаптивных блоках HTML-страниц

Начнем с самого простого — выравнивания текста по горизонтальной оси (`x`). Для этого нужно использовать класс `text`, который обеспечивает следующие варианты выравнивания:

- `start` (умолчание браузера) — по левой стороне;
- `end` — по правой стороне;
- `center` — по центру.

Вот пример горизонтального выравнивания текста в программном коде:

```
class="text-start"  
class="text-center"  
class="text-end"
```

Пример HTML-кода страницы, в которой показано выравнивание текста в теле адаптивных блоков, приведен в листинге 3.8, страница `alignment_1.html`.

Листинг 3.8. Страница `alignment_1.html`

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <title>alignment_1</title>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1">  
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >  
    <script defer src="static/js/bootstrap.bundle.min.js"></script>  
</head>  
<body>  
    <div class="container text-center">  
        <div class="row bg-success text-white">  
            <div class="col">Блок в первой строке</div>  
        </div>  
        <div class="row my-2 text-white">  
            <div class="col mx-2 bg-primary">Блок 1</div>  
            <div class="col mx-2 bg-primary">Блок 2</div>  
            <div class="col mx-2 bg-primary">Блок 3</div>  
        </div>  
    </div>
```

```
</div>
</body>
</html>
```

Здесь выравнивание текста задано в родительском классе `container (text-center)`. В этом случае во всех дочерних элементах текст будет выровнен по центру. В окне браузера данная страница будет иметь вид, представленный на рис. 3.18.

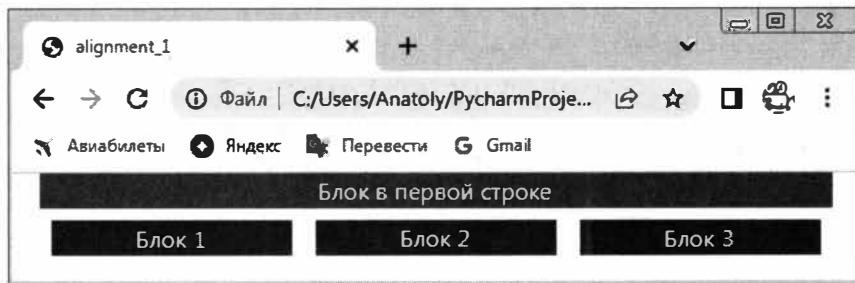


Рис. 3.18. Центрирование текста во всех аддаптивных блоках

Как видно из рис. 3.18, во всех дочерних элементах контейнера текст был выровнен по центру аддаптивного блока.

Немного изменим код этого примера и во втором ряду поменяем параметры выравнивания текста (листинг 3.9, страница `alignment_2.html`).

Листинг 3.9. Страница `alignment_2.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>alignment_2</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container text-center">
    <div class="row bg-success text-white">
        <div class="col">Блок в первой строке</div>
    </div>
    <div class="row my-2 text-white">
        <div class="col mx-2 bg-primary text-start">Блок 1</div>
        <div class="col mx-2 bg-primary">Блок 2</div>
        <div class="col mx-2 bg-primary text-end">Блок 3</div>
    </div>
</div>
</body>
</html>
```

Здесь выравнивание текста по центру задано в родительском классе `container (text-center)`. Однако во второй строке для первого блока задано выравнивание по левому краю (`text-start`), а для третьего блока — по правому краю (`text-end`). В окне браузера эта страница будет иметь вид, представленный на рис. 3.19.

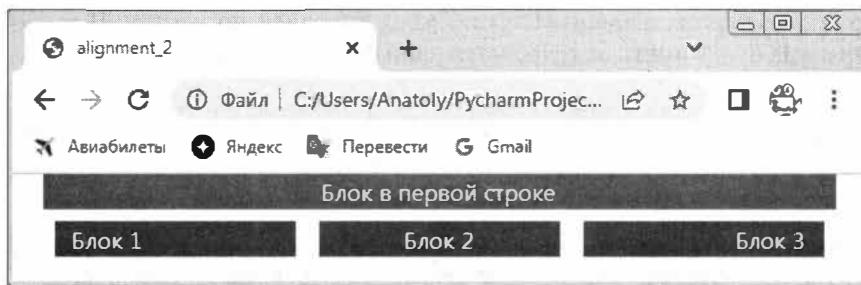


Рис. 3.19. Варианты выравнивания текста в адаптивных блоках

Как видно из данного рисунка, в адаптивных блоках текст выровнен по-разному: по центру, по левому и по правому краю.

3.9. Обозначение границ элементов макета HTML-страниц

В Bootstrap есть утилиты для формирования границ элементов макета, которые позволяют создать рамку, радиус закругления рамки, толщину и цвет рамки. Рамки отлично подходят для явного выделения блоков, для изображений, кнопок или любых других элементов интерфейса. Рамка может обрамлять элемент полностью или только одну из его сторон (рис. 3.20).

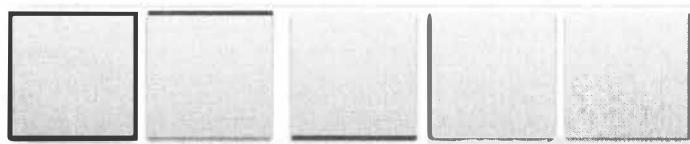


Рис. 3.20. Различные варианты обозначения границы элемента

Для добавления границ применяются следующие классы:

- `class="border"` — граница с четырех сторон;
- `class="border-top"` — граница сверху;
- `class="border-end"` — граница справа;
- `class="border-bottom"` — граница снизу;
- `class="border-start"` — граница слева.

Ширину рамки можно задать с помощью следующих классов:

- `class="border border-1";`
- `class="border border-2";`
- `class="border border-3";`

- class="border border-4";
- class="border border-5".

Пример HTML-кода страницы, в которой показано создание границ вокруг адаптивных блоков, приведен в листинге 3.10, страница border_1.html.

Листинг 3.10. Страница border_1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>border_1</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container text-center bg-primary">
    <div class="row bg-success text-white">
        <div class="col">Обозначение границ вокруг адаптивных блоков</div>
    </div>
    <div class="row my-2">
        <div class="col mx-2 bg-info border border-5">Блок 1</div>
        <div class="col mx-2 bg-info border-top border-5">Блок 2</div>
        <div class="col mx-2 bg-info border-bottom border-5">Блок 3</div>
        <div class="col mx-2 bg-info border-start border-5">Блок 4</div>
        <div class="col mx-2 bg-info border-end border-5">Блок 5</div>
    </div>
    <div class="row"></div>
</div>
</body>
</html>
```

Здесь созданы границы вокруг адаптивных блоков с разных сторон; цвет линии задан по умолчанию, а толщина — border-5. В окне браузера данная страница будет иметь вид, представленный на рис. 3.21.

Как видно из рис. 3.21, адаптивные блоки имеют ограничительные линии, расположенные с разных сторон.

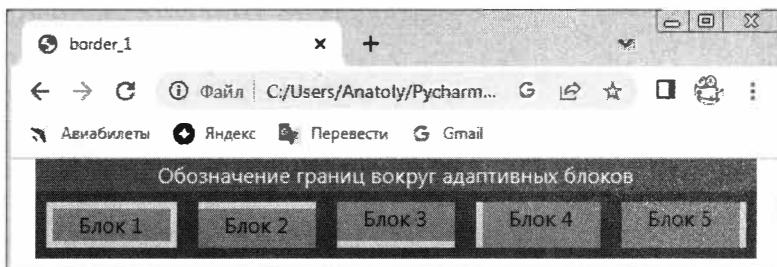


Рис. 3.21. Различные варианты обозначения границы вокруг адаптивного блока

Есть еще один способ обрамления адаптивных блоков, когда можно убирать границу с одной из сторон. Для этих целей используется следующий класс:

- class="border-0" — убирается ограничительная линия с четырех сторон;
- class="border-top-0" — убирается ограничительная линия сверху;
- class="border-end-0" — убирается ограничительная линия справа;
- class="border-bottom-0" — убирается ограничительная линия снизу;
- class="border-start-0" — убирается ограничительная линия слева.

Пример HTML-кода страницы, в которой показано создание границ вокруг адаптивных блоков путем удаления ограничительной линии с одной из сторон, приведен в листинге 3.11, страница border_2.html.

Листинг 3.11. Страница border_2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>border_2</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
    <div class="container text-center bg-primary">
        <div class="row bg-success text-white">
            <div class="col">Обозначение границ вокруг адаптивных блоков</div>
        </div>
        <div class="row my-2">
            <div class="col mx-2 bg-info border border-0">Блок 1</div>
            <div class="col mx-2 bg-info border border-5 border-top-0">Блок 2</div>
            <div class="col mx-2 bg-info border border-5 border-end-0">Блок 3</div>
            <div class="col mx-2 bg-info border border-5 border-bottom-0">
                Блок 4</div>
            <div class="col mx-2 bg-info border border-5 border-start-0">
                Блок 5</div>
        </div>
        <div class="row"></div>
    </div>
</body>
</html>
```

Здесь заданы границы вокруг адаптивных блоков с разных сторон. Цвет линии установлен по умолчанию, а толщина — border-5. В окне браузера эта страница будет иметь вид, представленный на рис. 3.22.

Как видно из данного рисунка, ограничительную линию можно удалить либо со всех сторон адаптивного блока, либо с одной из сторон.

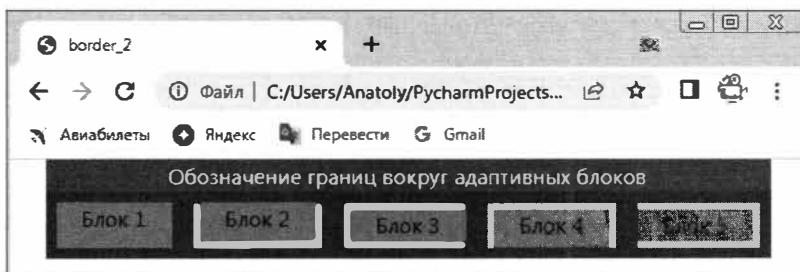


Рис. 3.22. Различные варианты обозначения границы вокруг адаптивного блока путем удаления ограничительной линии с одной из сторон

Изменить цвет границы вокруг адаптивного блока можно с помощью следующих классов:

- | | |
|---|---|
| <input type="checkbox"/> class="border border-primary"; | <input type="checkbox"/> class="border border-info"; |
| <input type="checkbox"/> class="border border-secondary"; | <input type="checkbox"/> class="border border-light"; |
| <input type="checkbox"/> class="border border-success"; | <input type="checkbox"/> class="border border-dark"; |
| <input type="checkbox"/> class="border border-danger"; | <input type="checkbox"/> class="border border-muted"; |
| <input type="checkbox"/> class="border border-warning"; | <input type="checkbox"/> class="border border-white". |

Пример HTML-кода страницы, в которой показано создание границ вокруг адаптивных блоков разного цвета, приведен в листинге 3.12, страница border_color.html.

Листинг 3.12. Страница border_color.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>border_1</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container text-center">
    <div class="row bg-success text-white">
        <div class="col">Обозначение границ вокруг адаптивных блоков</div>
    </div>
    <div class="row my-2">
        <div class="col mx-2 bg-info border border-5 border-primary">
            Блок 1</div>
        <div class="col mx-2 bg-info border border-5 border-secondary">
            Блок 2</div>
        <div class="col mx-2 bg-info border border-5 border-success">
            Блок 3</div>
        <div class="col mx-2 bg-info border border-5 border-danger">
            Блок 4</div>
    </div>
</div>
```

```

<div class="col mx-2 bg-info border border-5 border-warning">
    Блок 5</div>
<div class="col mx-2 bg-primary border border-5 border-info">
    Блок 6</div>
<div class="col mx-2 bg-info border border-5 border-light">Блок 7</div>
<div class="col mx-2 bg-info border border-5 border-dark">Блок 8</div>
<div class="col mx-2 bg-info border border-5 border-muted">Блок 9</div>
<div class="col mx-2 bg-info border border-5 border-white">Блок 10</div>
</div>
<div class="row"></div>
</div>
</body>
</html>

```

Здесь заданы цвета рамок границы вокруг адаптивных блоков с разных сторон толщиной линии border-5. В окне браузера данная страница будет иметь вид, представленный на рис. 3.23.

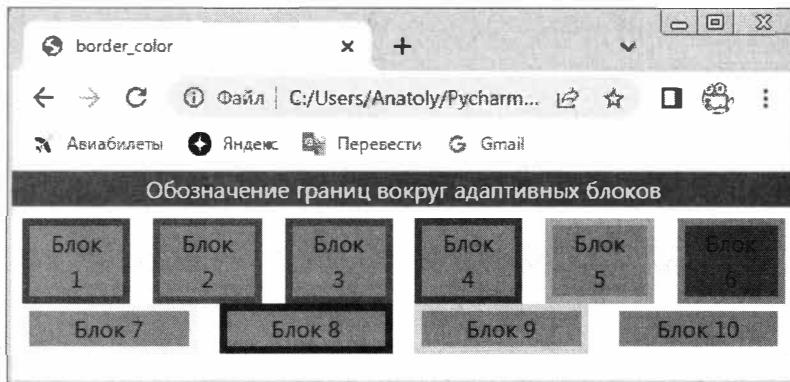


Рис. 3.23. Различные варианты цвета границы вокруг адаптивных блоков

Как видно из рис. 3.23, для ограничительных рамок можно задавать разные цвета. Обратите внимание на блоки 7 и 10, на которых не видно ограничительных рамок. Это связано с тем, что ограничительные рамки имеют тот же цвет, что и фон.

Углы ограничительных рамок можно округлить, для этого предусмотрены следующие классы:

- class="rounded";
- class="rounded-top";
- class="rounded-end";
- class="rounded-bottom";
- class="rounded-start";
- class="rounded-circle";
- class="rounded-pill".

Пример HTML-кода страницы, в которой показано создание рамок с округленными углами, приведен в листинге 3.13, страница border_radius.html.

Листинг 3.13. Страница border_radius.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>border_radius</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container text-center">
    <div class="row bg-success text-white">
        <div class="col">Обозначение границ вокруг адаптивных блоков</div>
    </div>
    <div class="row my-2">
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded">Блок 1</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-top">Блок 2</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-end">Блок 3</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-bottom">Блок 4</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-start">Блок 5</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-circle">Блок 6</div>
        <div class="col mx-2 bg-warning border border-3
            border-primary rounded-pill">Блок 7</div>
    </div>
    <div class="row"></div>
</div>
</body>
</html>
```

Здесь заданы рамки вокруг адаптивных блоков с толщиной линии border-3, при этом реализованы разные варианты округленных углов. В окне браузера данная страница будет иметь вид, представленный на рис. 3.24.

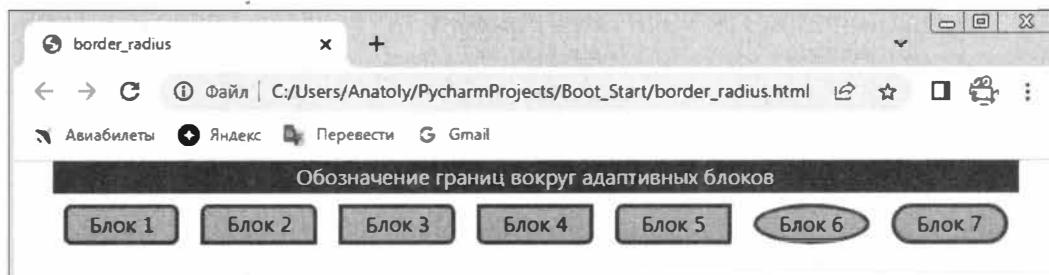


Рис. 3.24. Различные варианты округленных углов рамок вокруг адаптивного блока

Можно задать радиус округления углов ограничительных рамок, для этого используются следующие классы:

- class="rounded-0";
- class="rounded-1";
- class="rounded-2";
- class="rounded-3";
- class="rounded-4";
- class="rounded-5".

Пример HTML-кода страницы, в которой показано создание рамок с разными радиусами округления углов, приведен в листинге 3.14, страница border_radius_1.html.

Листинг 3.14. Страница border_radius_1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>border_radius_1</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container text-center">
    <div class="row bg-success text-white">
        <div class="col">Обозначение границ вокруг адаптивных блоков</div>
    </div>
    <div class="row my-2">
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-0">Блок 1</div>
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-1">Блок 2</div>
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-2">Блок 3</div>
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-3">Блок 4</div>
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-4">Блок 5</div>
        <div class="col mx-2 bg-warning border border-3 border-primary rounded-5">Блок 6</div>
    </div>
    <div class="row"></div>
</div>
</body>
</html>
```

Здесь созданы рамки вокруг адаптивных блоков с толщиной линии border-3, при этом заданы разные радиусы округленных углов. В окне браузера данная страница будет иметь вид, представленный на рис. 3.25.

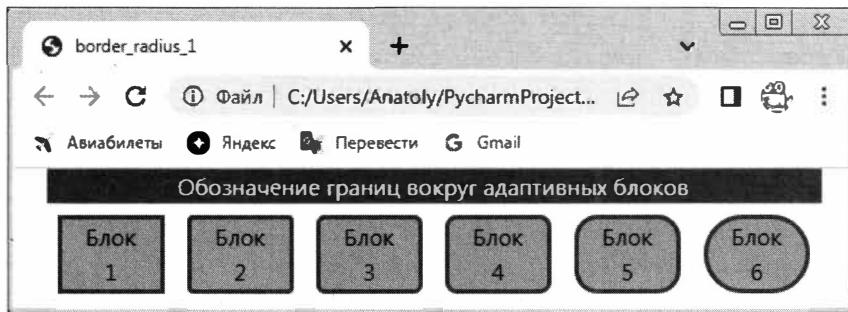


Рис. 3.25. Различные варианты радиусов округления углов рамок вокруг адаптивного блока

3.10. Пример использования адаптивных контейнеров

На HTML-странице может быть несколько адаптивных контейнеров. Для создания адаптивного контейнера с фиксированной шириной предусмотрен класс `container`. В качестве примера создадим HTML-страницу с двумя адаптивными контейнерами с фиксированной шириной (листинг 3.15, страница `start.html`).

Листинг 3.15. Страница `start.html`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Start</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container p-3 bg-primary text-white text-center">
    <h1>Это страница Bootstrap</h1>
    <p>Измените размер этой адаптивной страницы, чтобы увидеть эффект!</p>
</div>
<div class="container mt-3 text-center border border-3">
    <div class="row">
        <div class="col-sm-4 ">
            <h3>Колонка 1</h3>
            <p>Это содержимое</p>
            <p>первой колонки</p>
        </div>
        <div class="col-sm-4 ">
            <h3>Колонка 2</h3>
            <p>Это содержимое</p>
            <p>второй колонки</p>
        </div>
    </div>
</div>
```

```
<div class="col-sm-4 ">
    <h3>Колонка 3</h3>
    <p>Это содержимое</p>
    <p>третьей колонки</p>
</div>
</div>
</body>
</html>
```

На этой странице имеются два контейнера класса `container`. В первом контейнере находится текст, во втором — создан один ряд, в котором имеются три аддаптивных блока с вложенным в них текстом. В окне браузера данная страница будет иметь вид, представленный на рис. 3.26.

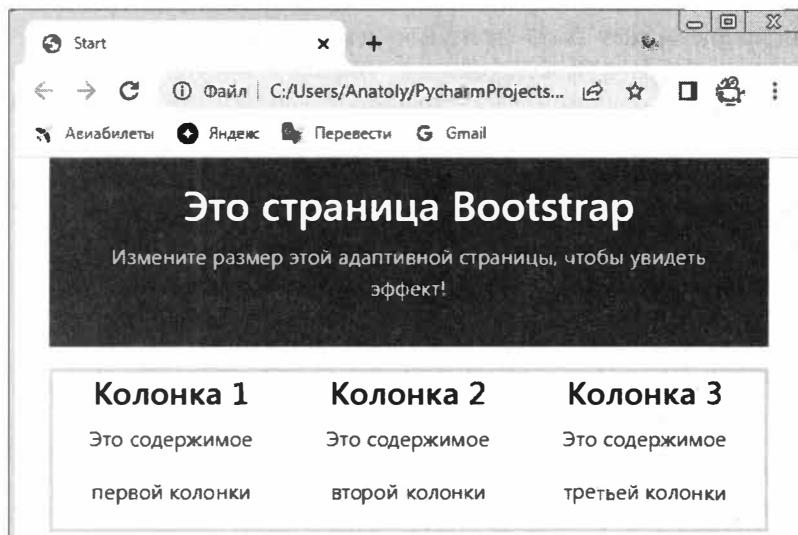


Рис. 3.26. Страница с двумя контейнерами с фиксированной шириной класса `container`

Обратите внимание, что при изменении ширины окна браузера меняется ширина контейнера и их содержимого, при этом окно браузера всегда шире, чем ширина контейнеров (слева и справа остается свободное пространство). При минимальной ширине окна браузера колонки трансформируются в три строки, и боковые отступы исчезают, при этом размер шрифта не уменьшается (рис. 3.27).

В таком виде страница будет отчетливо отображаться как на экране компьютера, так и на экране смартфона, текст будет оставаться разборчивым и читаемым.

Теперь создадим HTML-страницу с двумя аддаптивными подвижными, или «резиновыми» контейнерами на основе класса `container-fluid` (листинг 3.16, страница `start_1.html`).

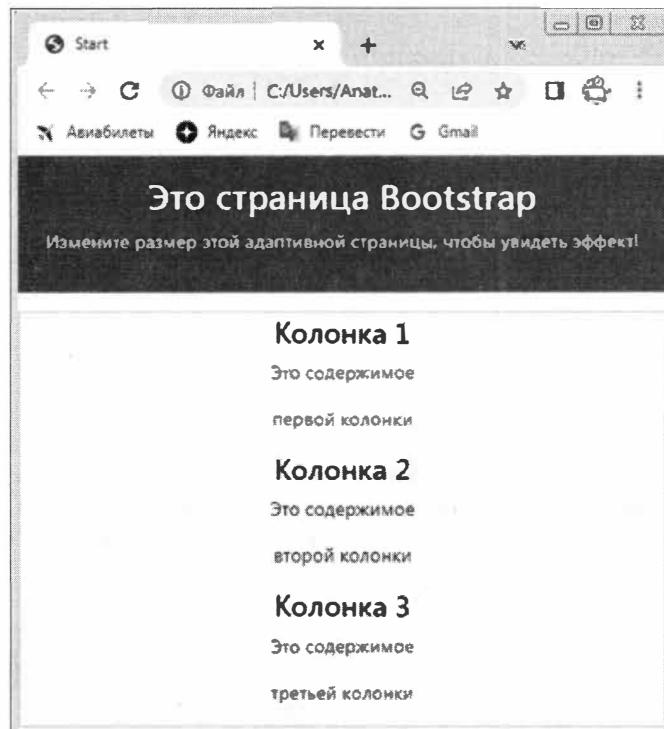


Рис. 3.27. Страница с двумя контейнерами с фиксированной шириной класса container при минимальном размере окна браузера

Листинг 3.16. Страница start_1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Start</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<div class="container-fluid p-3 bg-primary text-white text-center">
    <h1>Это страница Bootstrap</h1>
    <p>Измените размер этой адаптивной страницы, чтобы увидеть эффект!</p>
</div>

<div class="container-fluid mt-3 text-center border border-3">
    <div class="row">
        <div class="col-sm-4 ">
            <h3>Колонка 1</h3>
            <p>Это содержимое</p>
            <p>первой колонки</p>
        </div>
```

```
<div class="col-sm-4 ">
    <h3>Колонка 2</h3>
    <p>Это содержимое</p>
    <p>второй колонки</p>
</div>
<div class="col-sm-4 ">
    <h3>Колонка 3</h3>
    <p>Это содержимое</p>
    <p>третьей колонки</p>
</div>
</div>
</body>
</html>
```

На этой странице имеются два контейнера класса `container-fluid`. В первом контейнере находится текст, во втором — создан один ряд, в котором имеются три адаптивных блока с вложенным в них текстом. В окне браузера данная страница будет иметь вид, представленный на рис. 3.28.

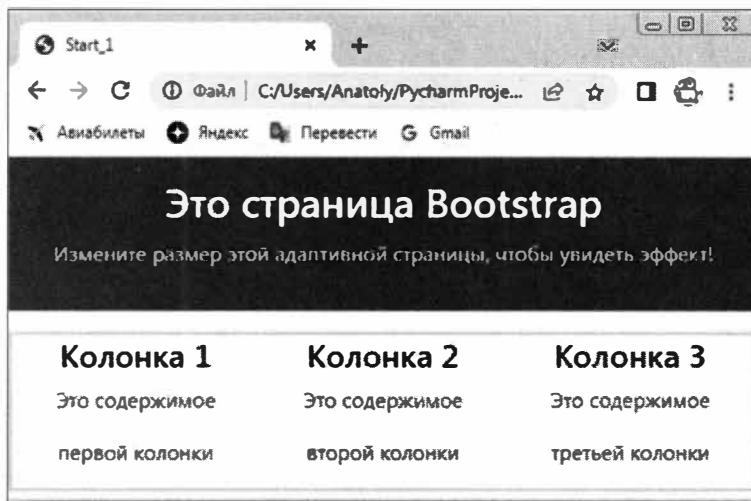


Рис. 3.28. Страница с двумя адаптивными подвижными контейнерами класса `container-fluid`

Обратите внимание, что при изменении ширины браузера меняется ширина контейнера и их содержимого, при этом ширина контейнеров всегда соответствует ширине окна браузера (слева и справа нет свободного пространства). При минимальной ширине браузера колонки трансформируются в три строки.

На этом мы закончим знакомство с макетированием HTML-страниц на основе фреймворка Bootstrap. Конечно, в Bootstrap имеется множество различных визуальных элементов для создания привлекательного веб-интерфейса, с которыми вы можете познакомиться в специальной литературе. Некоторые из этих элементов мы будем применять в шаблонах HTML-страниц и более детально познакомимся с ними в следующих главах на конкретных примерах.

3.11. Таблицы Bootstrap

Таблицы Bootstrap можно использовать как для вывода данных, так и для макетирования страниц. В базовом варианте таблицы имеют небольшие отступы и горизонтальные разделители. Для того чтобы создать таблицу, понадобятся следующие теги:

```

<table class="table">           ← Таблица
    <thead>                  ← Заголовок таблицы
        <tr>                   ← Стока заголовка
            <th>...</th>          ← Колонка заголовка
            ...
            <th>...</th>          ← Колонка заголовка
        </tr>
    </thead>

    <tbody>                   ← Тело таблицы
        <tr>                   ← Стока в теле таблицы
            <td>...</td>          ← Колонка в теле таблицы
            ...
            <td>...</td>          ← Колонка в теле таблицы
        </tr>
    </tbody>

</table>

```

Таблицу создают с помощью следующих тегов:

- <table> — создает таблицу;
- <thead> — заголовок таблицы;
- <tbody> — тело таблицы;
- <tr> — строку;
- <td> — колонку.

В качестве примера создадим HTML-страницу с таблицей, состоящей из трех строк и трех колонок (листинг 3.17, страница table.html).

Листинг 3.17. Страница table.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Таблица</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<table class="table">
    <thead>

```

```

<tr>
    <th>№</th>
    <th>Фамилия</th>
    <th>Имя</th>
    <th>E-mail</th>
</tr>
</thead>
<tbody>
    <tr>
        <td>1</td>
        <td>Иванов</td>
        <td>Иван</td>
        <td>ivan@mail.ru</td>
    </tr>
    <tr>
        <td>2</td>
        <td>Петров</td>
        <td>Петр</td>
        <td>petr@mail.ru</td>
    </tr>
    <tr>
        <td>3</td>
        <td>Сидоров</td>
        <td>Семен</td>
        <td>sid@mail.ru</td>
    </tr>
</tbody>
</table>
</body>
</html>

```

Здесь нет контейнеров, и таблица находится непосредственно в теле HTML-страницы. В окне браузера данная страница будет иметь вид, представленный на рис. 3.29.

Как видно из данного рисунка, с параметрами по умолчанию таблица представлена в черно-белом цвете, имеет выделенный заголовок и горизонтальные линии, разделяющие строки.

The screenshot shows a web browser window with a title bar 'Таблица'. The address bar displays the path 'C:/Users/Anatoly/P...'. Below the address bar are several icons: 'Авиабилеты', 'Яндекс', 'Перевести', and 'G Gmail'. The main content area contains a table with the following data:

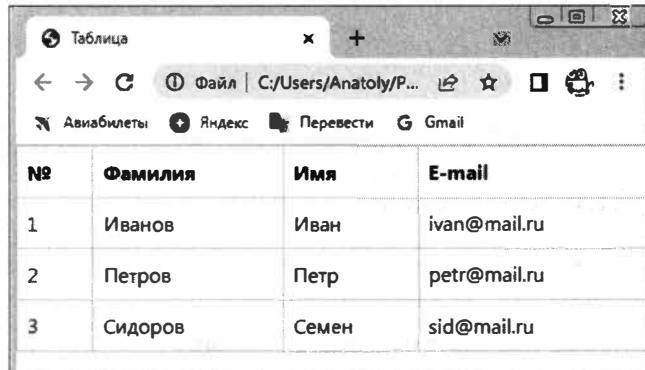
№	Фамилия	Имя	E-mail
1	Иванов	Иван	ivan@mail.ru
2	Петров	Петр	petr@mail.ru
3	Сидоров	Семен	sid@mail.ru

Рис. 3.29. Таблица Bootstrap

Для того чтобы создать для таблицы границы, используется класс `table-bordered`. Изменим в листинге 3.17 следующим образом строку с классом `table`:

```
<table class="table table-bordered">
```

После этого таблица примет вид, представленный на рис. 3.30.



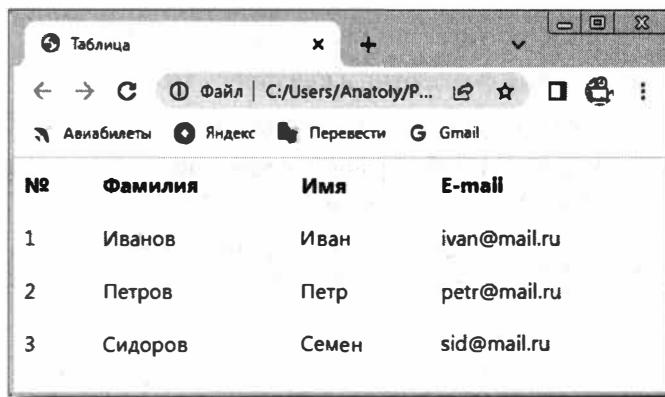
№	Фамилия	Имя	E-mail
1	Иванов	Иван	ivan@mail.ru
2	Петров	Петр	petr@mail.ru
3	Сидоров	Семен	sid@mail.ru

Рис. 3.30. Таблица Bootstrap с границами ячеек

Если таблица используется для макетирования страницы, то все границы можно удалить, для чего предусмотрен класс `table-borderless`. Для удаления границ изменим следующим образом строку с классом `table` в листинге 3.17:

```
<table class="table table-borderless">
```

После этого таблица примет вид, представленный на рис. 3.31.



№	Фамилия	Имя	E-mail
1	Иванов	Иван	ivan@mail.ru
2	Петров	Петр	petr@mail.ru
3	Сидоров	Семен	sid@mail.ru

Рис. 3.31. Таблица Bootstrap без границ ячеек

Считается хорошим тоном, когда четные и нечетные строки таблицы выделены альтернативными цветами. В Bootstrap это можно сделать с помощью класса `table-striped`. Изменим в листинге 3.17 следующим образом строку с классом `table`:

```
<table class="table table-striped">
```

После этого таблица примет вид, представленный на рис. 3.32.

№	Фамилия	Имя	E-mail
1	Иванов	Иван	ivan@mail.ru
2	Петров	Петр	petr@mail.ru
3	Сидоров	Семен	sid@mail.ru

Рис. 3.32. Таблица Bootstrap со строками альтернативного цвета

В таблицах можно использовать контекстные классы для окрашивания всей таблицы (`<table>`), строк таблицы (`<tr>`) или ячеек таблицы (`<td>`):

- `table-primary` — синий, указывает на важное действие;
- `table-success` — зеленый, указывает на успешное или положительное действие;
- `table-danger` — красный, указывает на опасное или потенциально негативное действие;
- `table-info` — голубой, указывает на нейтральное информативное изменение или действие;
- `table-warning` — оранжевый, указывает на предупреждение;
- `table-active` — серый, применяется при наведении курсора мыши на строку или ячейку таблицы;
- `table-secondary` — серый, указывает на менее важное действие;
- `table-light` — светло-серый фон таблицы или строки таблицы;
- `table-dark` — черный фон таблицы или строки таблицы.

В качестве примера создадим HTML-страницу с таблицей, в которой показаны классы с основными цветами (листинг 3.18, страница `table_1.html`).

Листинг 3.18. Страница `table_1.html`

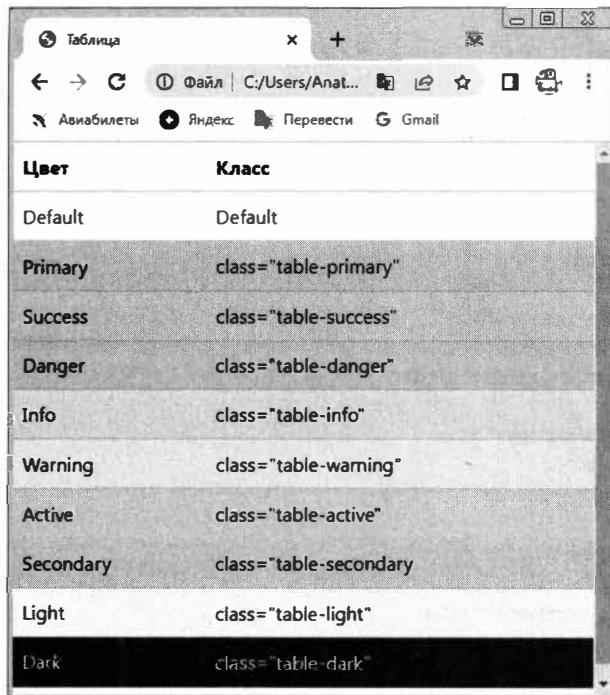
```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Таблица</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="static/css/bootstrap.min.css" >
    <script defer src="static/js/bootstrap.bundle.min.js"></script>
</head>
<body>
<table class="table">
    <thead>

```

```
<tr>
    <th>Цвет</th>
    <th>Класс</th>
</tr>
</thead>
<tbody>
    <tr>
        <td>Default</td>
        <td>Default</td>
    </tr>
    <tr class="table-primary">
        <td>Primary</td>
        <td>class="table-primary"</td>
    </tr>
    <tr class="table-success">
        <td>Success</td>
        <td>class="table-success"</td>
    </tr>
    <tr class="table-danger">
        <td>Danger</td>
        <td>class="table-danger"</td>
    </tr>
    <tr class="table-info">
        <td>Info</td>
        <td>class="table-info"</td>
    </tr>
    <tr class="table-warning">
        <td>Warning</td>
        <td>class="table-warning"</td>
    </tr>
    <tr class="table-active">
        <td>Active</td>
        <td>class="table-active"</td>
    </tr>
    <tr class="table-secondary">
        <td>Secondary</td>
        <td>class="table-secondary"</td>
    </tr>
    <tr class="table-light">
        <td>Light</td>
        <td>class="table-light"</td>
    </tr>
    <tr class="table-dark">
        <td>Dark</td>
        <td>class="table-dark"</td>
    </tr>
</tbody>
</table>
</body>
</html>
```

В окне браузера данная страница будет иметь вид, представленный на рис. 3.33.



The screenshot shows a web browser window titled 'Таблица' (Table). The address bar displays 'Файл | C:/Users/Anat...'. Below the title bar are standard browser buttons for back, forward, search, and refresh. The main content area contains a table with two columns: 'Цвет' (Color) and 'Класс' (Class). The rows are colored according to the Bootstrap color scheme:

Цвет	Класс
Default	Default
Primary	class="table-primary"
Success	class="table-success"
Danger	class="table-danger"
Info	class="table-info"
Warning	class="table-warning"
Active	class="table-active"
Secondary	class="table-secondary"
Light	class="table-light"
Dark	class="table-dark"

Рис. 3.33. Таблица Bootstrap со строками разного цвета

3.12. Краткие итоги

Итак, мы познакомились с некоторыми базовыми элементами фреймворка Bootstrap, которые нам пригодятся при формировании шаблонов HTML-страниц. Но прежде чем приступить к разработке веб-приложений с использованием Python, нужно познакомиться с фреймворком Django. В частности, нам предстоит узнать:

- историю появления фреймворка Django и его развитие;
- структуру веб-приложений на Django;
- механизм доступа к базам данных;
- механизм формирования динамических HTML-страниц на основе шаблонов.

И, наконец, получить практические навыки создания элементарных веб-приложений с помощью Django. Рассмотрению этих вопросов и посвящена следующая глава.