Интерактивная Электронная Книга (ИЭК)

Анатолий Постолит

Разработка web-приложений на Python, Django и Bootstrap

(интерактивное цифровое пособие)

Postolit Press

Постолит А. В.

Разработка web-приложений на Python, Django и Bootstrap (интерактивное цифровое пособие). — Postolit Press.: Москва, 2023. — 510 с.: ил.

ISBN 111-1-1111-1111-1

Интерактивная цифровая книга посвящена вопросам разработки веб-приложений с использованием языка Python, фреймворков Django, Bootstrap и интерактивной среды разработки PyCharm. Рассмотрены основные технологии и рабочие инструменты создания приложений, даны основы языка html. Приведено описание фреймворков Django, Bootsrtap и структура создаваемых веб-приложений. На простых примерах показана обработка и маршрутизация запросов пользователей, формирование ответных веб-страниц. Рассмотрено создание шаблонов веб-страниц и форм для пользователей. Показано взаимодействие пользователей с различными типами баз данных через модели. Описана работа с базами данных через встроенные в Django классы без использования SQL-запросов. Приведен пошаговый пример создания сайта от формирования шаблона до его администрирования и развертывания в сети Интернет с базами данных SQLite и MySQL. Уникальность интерактивной цифровой книг заключается в том, что в рамках одного web-приложения пользователь может просматривать текст книги, изучать листинги программного кода, запускать примеры программных модулей на выполнение, оценивать свои знания с помощью тестов, а также тестировать знания слушателей on-line курсов.

Оглавление

Предисловие

С развитием цифровых технологий и уходом в прошлое ряда популярных ранее специальностей молодые люди все чаще встают перед выбором перспективной, интересной и актуальной профессии. Международное интернет-сообщество считает, что одним из самых перспективных вариантов такого выбора является профессия веб-программиста. Именно эти специалисты формируют облик сети Интернет и создают технологические тренды будущего.

Каждую секунду в Интернете появляется от 3 до 5 новых сайтов, а каждую минуту — 80 новых интернет-пользователей. Все это технологическое "цунами" управляется разумом и умелыми руками веб-разработчиков. Зарплата этих специалистов вполне соответствует важности и актуальности их деятельности. Даже начинающие программисты на отечественном рынке могут рассчитывать на заработную плату от 50 тыс. рублей в месяц, а опытные программисты в России и за рубежом имеют доход, превышающий указанную цифру в десятки раз.

Специалисты IT-технологий уже много лет подряд занимают первые позиции в рейтингах кадровых агентств, как представители наиболее востребованных профессий на отечественном и мировом рынке труда. Постоянная нехватка квалифицированных программистов дает высокий шанс молодым и целеустремленным новичкам для входа в эту профессию.

Согласно данным Ассоциации предприятий компьютерных и информационных технологий (АПКИТ) спрос на IT-специалистов ежегодно остается на высоком уровне, однако кандидатов по-прежнему не хватает. В ближайшие несколько лет дефицит будет только нарастать, так как IT-специалисты нужны повсеместно. Если раньше ими интересовались сугубо профильные компании (разработчики софта и онлайн-платформ), то сейчас квалифицированные кадры IT-профилей требуются практически везде. И речь здесь идет не о дефиците, а о катастрофическом дефиците. Уже много лет подряд сервис по поиску работы SuperJob и hh.ru фиксирует огромный неудовлетворенный спрос на IT-специалистов.

Рассматриваемый в этой книге язык программирования Python — один из самых популярных языков программирования, и области его применения только расширяются. Последние несколько лет он входит в ТОП-3 самых востребованных языков, а в 2022 году вышел на первое место. Задействуя Python, можно решать множество научных проблем и задач в области бизнеса. К нему обращаются ученые (математики, физики, биологи), так как изучить его не слишком сложно. Он используется при реализации нейронных сетей, для написания веб-сайтов и мобильных приложений. В целом это универсальный язык, входящий в тройку языков для анализа больших данных. В течение последних 5 лет Python-разработчики востребованы на рынке труда, и специалистов в этой сфере до сих пор не хватает.

Еще одна причина, по которой следует обратить внимание на использование языка Python, фреймворков Django и Bootstrap для веб-разработки, — это развитие систем искусственного интеллекта. На Pytnon реализовано большое число библиотек, облегчающих создание программного обеспечения для систем искусственного интеллекта, Django обеспечивает разработку серверной части, а Bootstrap — клиентской части веб-приложений с развертыванием их в сети Интернет. С помощью этого инструментария можно создавать приложения для беспилотных автомобилей, для интеллектуальных транспортных систем, для телемедицины, систем обучения, распознавания объектов в системах безопасности и т. п. Это очень востребованные и актуальные сферы, где наблюдается еще большая потребность в специалистах. Здесь талантливые и целеустремленные молодые люди могут найти как возможность самореализации, так и достойную оплату своего труда.

Каждая организация, принимающая на работу IT-специалиста, требует знаний, которые будут полезны именно в ее работе. Однако общее правило таково, что чем больше популярных и необходимых языков программирования, фреймворков и баз данных вы знаете (Js, HTML, C#, C++, Python, PHP, Django, SQL) и чем больше ваш опыт работы, тем выше шансы на удачное трудоустройство и достойную зарплату.

Перед теми, кто решил освоить специальность веб-программиста, встает непростой выбор — с чего же правильно начать. Конечно, всегда существует возможность получить полноценное IT-образование в одном из ведущих технических вузов ранга МГУ, МГТУ им. Н. Баумана, СПбГУ, МФТИ, ИТМО. Но обучение в них обойдется в круглую сумму от 60 до 350 тыс. рублей в год. Существует и более быстрый и дешевый вариант стать веб-разработчиком "с нуля", пройдя краткосрочные онлайн-курсы. Однако практикующие программисты уверяют, что на начальном этапе самый правильный способ обучиться веб-программированию — освоить его самостоятельно. Так можно не только избежать серьезных расхо-

дов, но и получить только те практические навыки, которые пригодятся в будущей работе. Полученные самостоятельно базовые знания впоследствии можно расширить, обучаясь уже на соответствующих курсах или в специализированном вузе.

Эта книга предназначена как для начинающих программистов (школьников и студентов), так и для специалистов с опытом, которые планируют разрабатывать или уже занимаются разработкой веб-приложений с использованием Python. Сразу следует отметить, что веб-программирование требует от разработчиков бóльших знаний, умений и усилий, чем программирование традиционных приложений. Здесь, кроме основного языка, который реализует логику приложения, необходимо еще и знание структуры HTML-документов, основ работы с базами данных, принципов взаимодействия удаленных пользователей с веб-приложением. Если некоторые разделы книги вам покажутся трудными для понимания и восприятия, то не стоит отчаиваться. Нужно просто последовательно, по шагам повторить приведенные в книге примеры. А когда вы увидите результаты работы программного кода, появится ясность — как работает тот или иной элемент изучаемого фреймворка. Конечно, для тех, кто собирается заниматься созданием веб-приложений на основе Django, необходимы базовые знания Python: переменные, списки, кортежи, словари, функции, классы. Эти знания можно получить из специальной литературы.

В книге рассмотрены практически все элементарные действия, которые выполняют программисты, работая над реализацией веб-приложений, приведено множество примеров и проверенных программных модулей. Рассмотрены базовые классы фреймворка Django, методы и свойства каждого из классов и примеры их использования. Книга, как уже отмечалось, предназначена как для начинающих программистов, приступивших к освоению языка Python, так и специалистов, имеющих опыт программирования на других языках. В этом издании в меньшей степени освещены вопросы дизайна веб-приложений, а больше внимания уделено технологиям разработки веб-приложений на практических примерах. Тем не менее, в книге приводятся базовые сведения о фреймворке Bootstrap и его использование вместе с Django для макетирования и разработки дизайна HTML-страниц.

Первая глава книги посвящена инструментальным средствам, которые применяются для разработки веб-приложений. В ней сказано, как установить на локальный компьютер Python, среду для написания программного кода PyCharm, фреймворк Django, менеджер для работы с базами данных SQLiteStudio. Кроме того показано, как установить различные дополнительные пакеты в Python с использованием менеджера пакетов pip.

Вторая глава книги посвящена знакомству с веб-технологиями, рассмотрены принципиальные различия между приложениями, работающими на сервере и на стороне клиента. Показаны различия между такими понятиями, как *frontend-разработка* и *backend-разработка*. Приведены базовые сведения о HTML-страницах, их структуре и основных тегах языка HTML, позволяющих выводить и форматировать информацию. Представлен перечень и краткое описание языков программирования, позволяющих создавать веб-приложения, даны базовые сведения о каскадных таблицах стилей (CSS), и о JavaScript.

В *третьей главе* рассматриваются возможности макетирования HTML-страниц с помощью фреймворка Bootstrap. Из этой главы вы узнаете, как можно получить файлы фреймворка Bootstrap и подключить их к своему проекту, что такое контейнеры и сетка Bootstrap. На простых примерах показано, как можно разбить HTML-страницы на адаптивные блоки и придать им привлекательный внешний вид. Это очень полезный и достаточно популярный инструмент, который позволяет выполнить оформление внешнего вида сайта.

Четвертая глава посвящена основным понятиям и определениям, которые приняты в фреймворке Django. В ней также описана структура приложений на Django. С использованием интерактивной среды PyCharm мы создадим здесь первый простейший проект на Django и сформируем в этом проекте приложение.

В *пятой главе* приведены основные понятия о представлениях (view) и маршрутизации запросов пользователей. Это весьма важные компоненты, которые определяют, что должно делать веб-приложение при поступлении различных запросов от удаленных пользователей. Здесь же описан синтаксис так называемых *регулярных выражений*, которые преобразуют запросы пользователей в адреса перехода к тем или иным страницам сайта.

В *шестой главе* рассмотрены вопросы создания шаблонов HTML-страниц. На конкретных примерах показано, как передать в шаблоны простые и сложные данные. Приведены примеры использования статичных файлов в приложениях на Django. Рассмотрена возможность расширения шаблонов HTML-страниц на основе базового шаблона — это, по сути, основной прием, который применяется практически на всех сайтах. Приведены также примеры использования в шаблонах HTML-страниц специальных тегов (для вывода текущей даты и времени, вывода информации по условию, вывода информации в цикле и для задания значений переменным).

В седьмой главе сделан обзор пользовательских форм. Формы — это основной интерфейс, благодаря которому пользователи имеют возможность вносить информацию в удаленную базу данных. Из этой главы вы узнаете, как использовать в формах POST-запросы, в ней также подробно описаны поля, которые

можно задействовать для ввода данных, и приводятся короткие примеры применения каждого типа поля. Кроме того, рассмотрены примеры изменения внешнего вида полей с помощью виджетов (widget), настроек вида полей, валидации данных и стилизации полей на формах Django.

Восьмая глава посвящена изучению моделей Django. Модели, по своей сути, представляют собой описание таблиц и полей базы данных (БД), используемой веб-приложением. На основе моделей будут автоматически созданы классы, через свойства и методы которых приложение станет взаимодействовать с базой данных. Соответственно, через классы будут выполняться все процедуры работы с данными (добавление, модификация, удаление записей из таблиц БД), при этом отпадает необходимость в написании SQL-запросов — Django будет создавать их самостоятельно и задействовать при манипулировании данными. Кроме того, посредством процедуры миграции Django в автоматическом режиме создаст необходимые таблицы в различных СУБД (SQLite, PostgreSQL, MySQL, Oracle).

В *девятой главе* мы создадим модели для достаточно простого "учебного" сайта "Мир книг" и соответствующие таблицы в системе управления базами данных (СУБД) SQLite. Возможность работы с данными этого сайта здесь будет показана через встроенную в Django административную панель.

В *десятой главе* приведен пример создания веб-интерфейса для пользователей сайта "Мир книг". Здесь будет определен перечень страниц сайта и их интернет адреса (URL), созданы представления (views) для обработки запросов пользователей, базовый шаблон сайта и шаблоны прочих страниц сайта. Это простейшие страницы для получения информации из БД на основе запросов пользователей. Здесь к проекту Django будут подключены файлы Bootstrap и продемонстрировано их взаимодействие.

В *одиннадцатой главе* показано, как можно расширить возможности администрирования сайта "Мир книг" и обеспечить ввод и редактирование данных со стороны удаленных пользователей через формы Django. Здесь рассмотрено такое понятие, как *сессия*, представлены процедуры создания страниц авторизация пользователей, проверки подлинности входа пользователей в систему, изменение внешнего вида страниц для зарегистрированных пользователей. Подробно на примерах показано применение классов `Form` и `ModelForm`.

Двенадцатая глава посвящена теме публикации сайтов, разработанных на Django, в сети Интернет. Эта, казалось бы, несложная процедура требует от разработчика определенной квалификации и навыков. На удаленном сервере потребуется создать соответствующее окружение для Python, подгрузить необходимые библиотеки и модули, подключить нужную СУБД, настроить пути к статичным файлам рисунков и стилей. В качестве примера мы выполним процедуру публикации "учебного" сайта "Мир книг" на российском сетевом ресурсе timeweb с использованием СУБД SQLite. В этой же главе вы найдете информацию о том, как зарегистрировать доменное имя и подключить его к сайту, опубликованному на публичном сервере.

В *тринадцатой главе* показано, как можно переключить разработанное приложение с СУБД SQLite на MySQL. Из этой главы вы узнаете, как установить сервер MySQL на локальный компьютер, создать базу данных на MySQL и подключить к ней проект Django. Так же будет рассказано о том, как можно создать инфраструктуру на удаленном сервере для развертывания сайта Django с СУБД MySQL и перенести свой сайт с локального компьютера на публичный сервер.

На протяжении всей книги раскрываемые вопросы сопровождаются достаточно упрощенными, но полностью законченными примерами. Ход решения той или иной задачи показан на большом количестве иллюстративного материала. Желательно изучение тех или иных разделов осуществлять, сидя непосредственно за компьютером, — тогда вы сможете последовательно повторять выполнение тех шагов, которые описаны в примерах, и тут же видеть результаты своих действий, что в значительной степени облегчит восприятие материала книги. Наилучший способ обучения — это практика. Все листинги программ приведены на языке Python, а шаблоны веб-страниц — в виде HTML-файлов. Это прекрасная возможность познакомиться с языком программирования Python и понять, насколько он прост и удобен в использовании.

Вы будете работать не с бумажной версией книги, а с ее цифровым аналогом. Уникальность и преимущество изучения программирования с использованием интерактивных цифровых книг заключается в том, что вы в рамках одно web-приложения может просматривать текст книги, изучать листинги программного кода и запускать примеры программных модулей на выполнение. Для этого нет необходимости устанавливать на свой компьютер набор инструментальных средств, так как весь инструментарий, нужные библиотеки и отлаженные примеры программного кода уже загружены на удаленный сервер. Это избавит от необходимости тратить время на ручной набор программного кода и его отладку.

Следует отметить еще одно важное преимущество цифровых книг перед бумажными книгами – все рисунки в цифровой книге цветные. Цена бумажных книг с цветными рисунками в два-три раза превышает цену аналогичной книги с черно-белыми рисунками (это связано с более высокими затратами типографий на цветную печать), что отрицательно сказывается на их покупательной способности. Поэтому большинство издательств, повышая покупательную привлекательность за счет более низкой цены, издают книги в черно-белом варианте. Однако книги по программированию, особенно связанные с инстру-

ментарием для разработки пользовательского интерфейса, должны издаваться в цветном варианте, в противном случае в них невозможно показать, например, элементы дизайна веб-интерфейса.

Цифровые книги имеют преимущества перед бумажными книгами: все изображения можно показать в цвете (это никак не скажется на стоимости издания), тираж таких книг не имеет ограничений, затраты на обновление и переиздание цифровых книги в десятки раз меньше, чем бумажных. Цифровые книги можно продавать через сеть интернет, а это более привлекательный способ распространения, чем через склады и сеть книжных магазинов. Стоимость цифровых книг в полтора раза меньше, чем бумажных, что благоприятно сказывается на их покупательной способности.

При изучении материалов интерактивной цифровой книги есть возможность, двигаясь по оглавлению, практически мгновенно открыть и просмотреть любой раздел в любой главе книги. Кроме того, можно открыть разные разделы текст книги в нескольких вкладках и оперативно переключаться между несколькими разделами. Пользователь может не только детально изучить тот или иной фрагмент программного кода, но и, запустив его на выполнение, увидеть результаты работы того или иного примера, скачать программный код на свой компьютер и использовать в своих проектах.

По мере освоения того или иного раздела в раках интерактивной цифровой книги, пользователь может формировать на своем рабочем компьютере собственную среду разработки, и уже в ней самостоятельно практиковаться, создавая учебные или реальные приложения. При этом интерактивная цифровая книга будет тем справочным материалом, к которому можно будет постоянно обращаться по мере возникновения тех или иных вопросов. Самое ценное в любой книге по программированию – это доступный, хорошо структурированный и отлаженный программный код, а именно такой код содержится в интерактивной цифровой книге.

Интерактивная цифровая книга представляет собой веб-приложение. Инструкция, в которой описан порядок работы с интерактивной цифровой книгой, приведена в приложении.

Итак, если вас заинтересовали вопросы создания веб-приложений с использованием Python, Django и Bootstrap в среде PyCharm, то самое время перейти к изучению материалов этой книги.

Глава 1. Инструментальные средства для разработки веб-приложений

Для разработки веб-приложений существует множество языков программирования, каждый из которых имеет свои особенности. Следует отметить, что веб-приложения состоят из двух частей — клиентской (frontend) и серверной (backend). Исходя из этого, программистов, которые занимаются веб-приложениями, делят на две категории: frontend-разработчики и backend-разработчики, и они используют разные языки программирования. Для разработки клиентской части веб-приложения применяют язык гипертекстовой разметки HTML в сочетании с CSS и JavaScript. А вот серверную часть можно разрабатывать с помощью совершенно разных языков программирования: PHP, Python, Ruby, C#, Perl. Но из них хочется выделить Python, который стал наиболее распространенным и универсальным средством разработки программного кода на протяжении уже более тридцати лет.

Python — интерпретируемый язык программирования, который в конце 1989 года создал Голландский программист Гвидо Ван Россум, и он очень быстро стал популярным и востребованным у программистов. В подтверждение этого можно упомянуть компании-гиганты: Google, Microsoft, Facebook, Yandex и многие другие, которые используют Python для реализации глобальных проектов.

Область применения Python очень обширна — это и обработка научных данных, и системы управления жизнеобеспечением, а также игры, веб-ресурсы, системы искусственного интеллекта. За все время существования Python динамично развивался. Для него создавались стандартные библиотеки, обеспечивающие поддержку современных технологий — например, для работы с базами данных, протоколами Интернета, электронной почтой, машинным обучением и многим другим.

Ускорить процесс написания программного кода позволяет специализированная инструментальная среда — так называемая *интегрированная среда разработки* (IDE, Integrated Development Environment). Эта среда включает полный комплект средств, необходимых для эффективного программирования на Python. Обычно в состав IDE входят текстовый редактор, компилятор или интерпретатор, отладчик и другое программное обеспечение. Применение IDE позволяет увеличить скорость разработки программ (при условии предварительного обучения работе с такой инструментальной средой). На сегодняшний день наиболее популярная среда для разработки приложений на Python — IDE PyCharm. Именно это программное обеспечение будет использовано при написании программного кода для данной книги.

Возможность создавать веб-приложения не встроена в основные функции Python. Для реализации задач подобного класса на Python нужен дополнительный инструментарий. Таким инструментарием является фреймворк Django, который считается лучшим фреймворком, написанным на Python, предназначенным для написания серверной части веб-приложений. Этот инструмент хорошо подходит для создания сайтов, работающих с базами данных, он делает веб-разработку на Python очень качественной и удобной.

Что касается разработки клиентской части веб-приложений, то в настоящее время наиболее популярным инструментом считается фреймворк Bootstrap. С его помощью достаточно просто и удобно создавать пользовательский интерфейс, который адаптируется под экраны разного размера и разрешения. Страницы сайтов, созданные на основе Bootstrap, хорошо читаются и имеют привлекательный вид на больших экранах настольных компьютеров, на средних экранах ноутбуков и планшетов, на малых экранах смартфонов. Вместе с Django можно использовать как оригинальный фреймворк Bootstrap, так и его различные адаптированные версии, например, django-bootstrap-v5. В данной книге нам понадобятся лишь некоторые элементы фреймворка Bootstrap, предназначенные для макетирования HTML страниц.

Большинство современных веб-приложений обеспечивают взаимодействие пользователей с базами данных, в которых хранятся сведения о зарегистрированных пользователях, тексты, изображения, аудио и видео контент. Доступ к содержимому баз данных осуществляется с помощью языка SQL. Несмотря на то, что Django позволяет обойти прямое использование SQL через модели данных, знание основ SQL важно для веб-разработки. Приложения на Django могут взаимодействовать с различными базами данных, а по умолчанию принята простейшая система управления базами данных (СУБД) SQLite. В процессе работы над примерами книги необходимо будет заглядывать в содержимое таблиц СУБД, а для этого нужно иметь соответствующее программное средство. Работать с СУБД SQLite мы будем посредством простейшего менеджера баз данных SQLiteStudio.

Из материалов этой главы вы также узнаете, как установить следующие компоненты:

- интерпретатор Python;

- интерактивную среду разработки программного кода PyCharm;
- различные дополнительные пакеты в Python с использованием менеджера пакетов pip;
- фреймворк Django.
- менеджер баз данных SQLiteStudio.

Примечание.
Для работы с интерактивной цифровой книгой вам не нужно инсталлировать на свой компьютер никаких дополнительных инструментальных средств. Все необходимой программное обеспечение уже установлено на публичном сервере, и, работая с сайтом через интернет браузер, вы сможете запустить на выполнение код всех примеров книги из глав 2-8. Однако, начиная с главы 9, вы начнете уже самостоятельно создавать простейшее веб-приложение, для чего нужно будет сформировать на своем компьютере соответствующую инструментальную среду.

1.1. Интерпретатор Python

Язык программирования Python — это весьма мощное инструментальное средство для разработки различных систем. Однако наибольшую ценность представляет даже не столько сам язык программирования, сколько набор подключаемых библиотек, на уровне которых уже реализованы все необходимые процедуры и функции. Разработчику достаточно написать несколько десятков строк программного кода, чтобы подключить требуемые библиотеки, создать набор необходимых объектов, передать им исходные данные и отобразить итоговые результаты.

Для установки интерпретатора Python на компьютер, прежде всего надо загрузить его дистрибутив. Скачать дистрибутив Python можно с официального сайта, перейдя по ссылке **https://www.python.org/downloads/** (рис. 1.1).

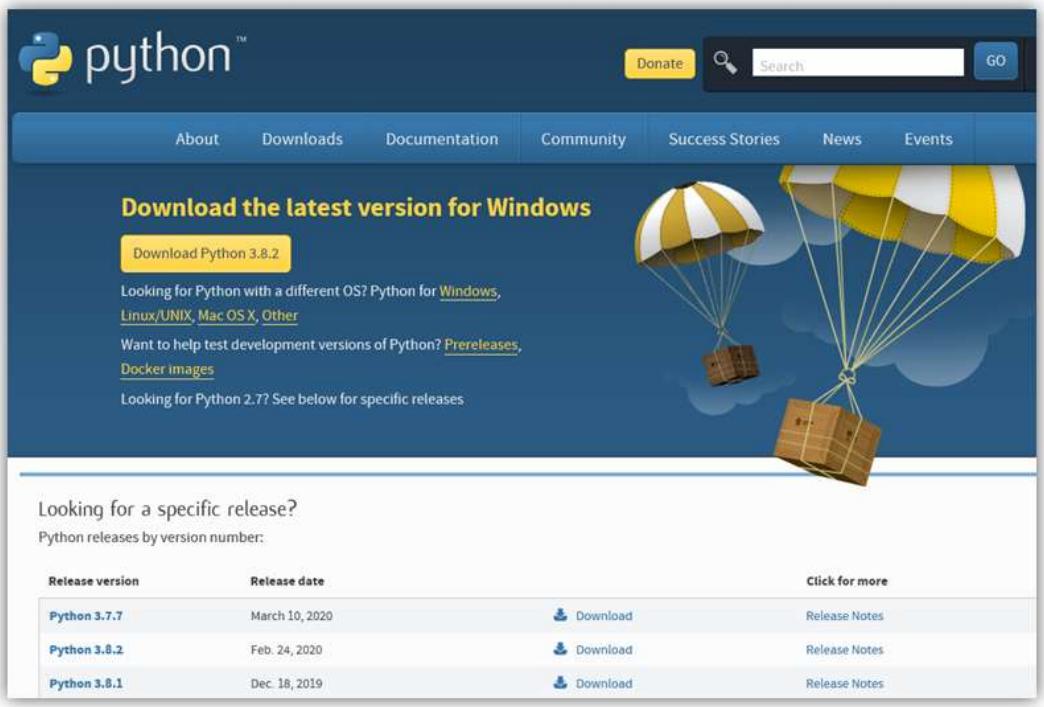

Рис. 1.1. Сайт для скачивания дистрибутива языка программирования Python

На момент написания книги доступен Python версии 3.11, однако, здесь мы будем использовать более раннюю версию 3.8, поскольку она стабильно работает на всех версиях Windows, начиная с Windows 7 и выше.

1.1.1. Установка Python в Windows

Для операционной системы Windows дистрибутив Python распространяется либо в виде исполняемого файла (с расширением exe), либо в виде архивного файла (с расширением zip). При написании программного кода для этой книги был использован Python версии 3.8.3.

Порядок установки Python в Windows следующий:

1. Запустите скачанный установочный файл.

2. Выберите способ установки (рис. 1.2).

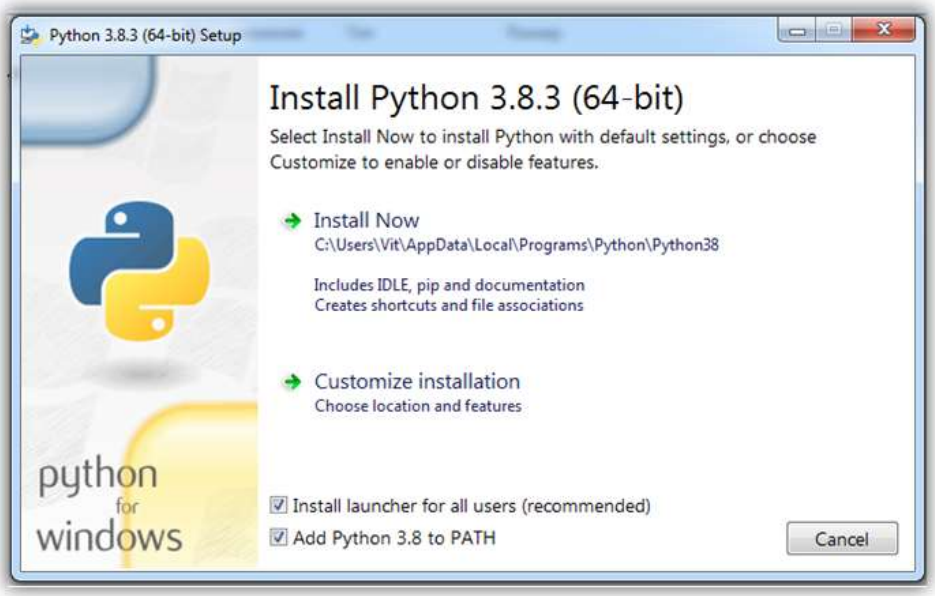

Рис. 1.2. Выбор способа установки Python

В открывшемся окне предлагаются два варианта: **Install Now** и **Customize installation**:

- при выборе **Install Now** Python установится в папку по указанному в окне пути. Помимо самого интерпретатора будут инсталлированы IDLE (интегрированная среда разработки), pip (пакетный менеджер) и документация, а также созданы соответствующие ярлыки и установлены связи (ассоциации) файлов, имеющих расширение py, с интерпретатором Python;
- **Customize installation** — это вариант настраиваемой установки. Опция **Add Python 3.8 to PATH** нужна для того, чтобы появилась возможность запускать интерпретатор без указания полного пути до исполняемого файла при работе в командной строке.

3. Отметьте необходимые опции установки, как показано на рис. 1.3 (доступно при выборе варианта **Customize installation**).

На этом шаге нам предлагается отметить дополнения, устанавливаемые вместе с интерпретатором Python. Рекомендуется выбрать как минимум следующие опции:

- **Documentation** — установка документации;
- **pip** — установка пакетного менеджера pip;

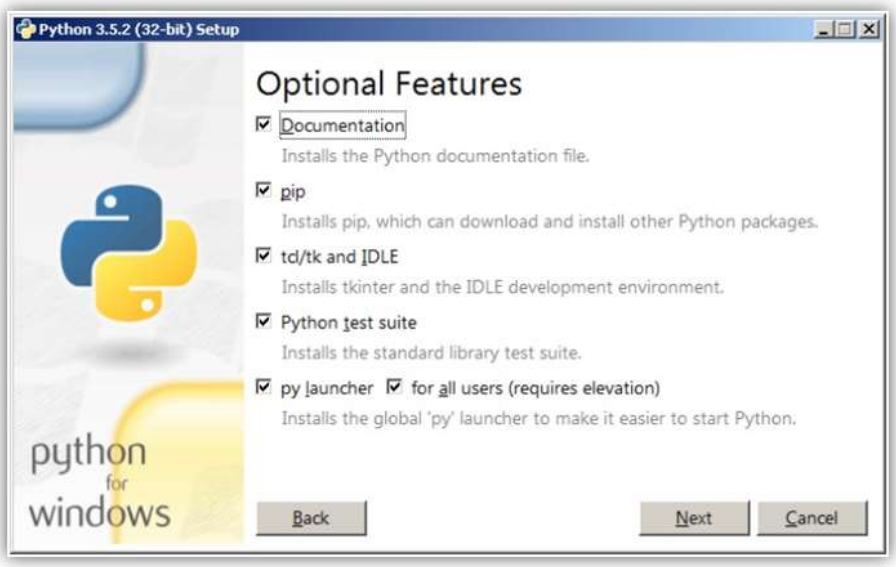

Рис. 1.3. Выбор опций установки Python

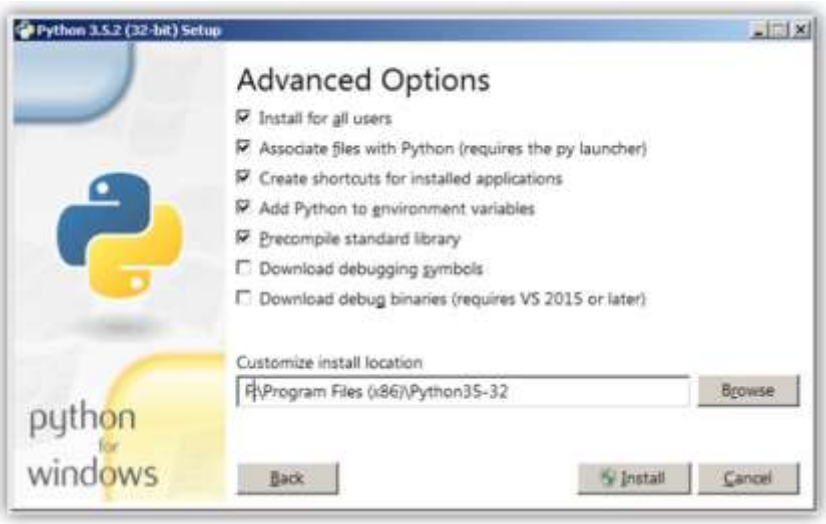

Рис. 1.4. Выбор места установки Python

- **tcl/tk and IDLE** — установка интегрированной среды разработки (IDLE) и библиотеки для построения графического интерфейса (tkinter).

4. На следующем шаге в разделе **Advanced Options** (дополнительные опции) выберите место установки, как показано на рис. 1.4 (доступно при выборе варианта **Customize installation**).

   Помимо указания пути, этот раздел позволяет внести дополнительные изменения в процесс установки с помощью опций:

   - **Install for all users** — установить для всех пользователей. Если не выбрать эту опцию, то будет предложен вариант инсталляции в папку пользователя, устанавливающего интерпретатор;
   - **Associate files with Python** — связать файлы, имеющие расширение py, с Python. При выборе этой опции будут внесены изменения в Windows, позволяющие Python запускать скрипты по двойному щелчку мыши;
   - **Create shortcuts for installed applications** — создать ярлыки для запуска приложений;
   - **Add Python to environment variables** — добавить пути до интерпретатора Python в переменную PATH;
   - **Precomple standard library** — провести перекомпиляцию стандартной библиотеки.

   Последние два пункта (см. рис.1.4) связаны с загрузкой компонентов для отладки, их мы устанавливать не будем.

5. После установки Python вас ждет следующее сообщение об успешном завершении установки (рис. 1.5).

Рис. 1.5. Финальное сообщение после установки Python

1.1.2. Установка Python в Linux

Чаще всего интерпретатор Python уже входит в состав дистрибутива Linux. Это можно проверить, набрав в окне терминала команду:

```
> python
```

или

```
> python3
```

В первом случае вы запустите Python 2, во втором — Python 3. В будущем, скорее всего, во все дистрибутивы Linux, включающие Python, будет входить только третья его версия. Если у вас при попытке запустить Python выдается сообщение о том, что он не установлен или установлен, но не тот, который вам необходим, то у вас есть возможность взять его из репозитория.

Для установки Python из репозитория Ubuntu воспользуйтесь командой:

```
> sudo apt-get install python3
```

1.1.3. Проверка интерпретатора Python

Для начала протестируем интерпретатор в командном режиме. Если вы работаете в Windows, то нажмите комбинацию клавиш <Win> + <R> и в открывшемся окне введите: `python`. В Linux откройте окно терминала и в нем введите: `python3` (или `python`).

В результате Python запустится в командном режиме. Выглядеть это будет примерно так, как показано на рис. 1.6 (иллюстрация приведена для Windows, в Linux результат будет аналогичным).

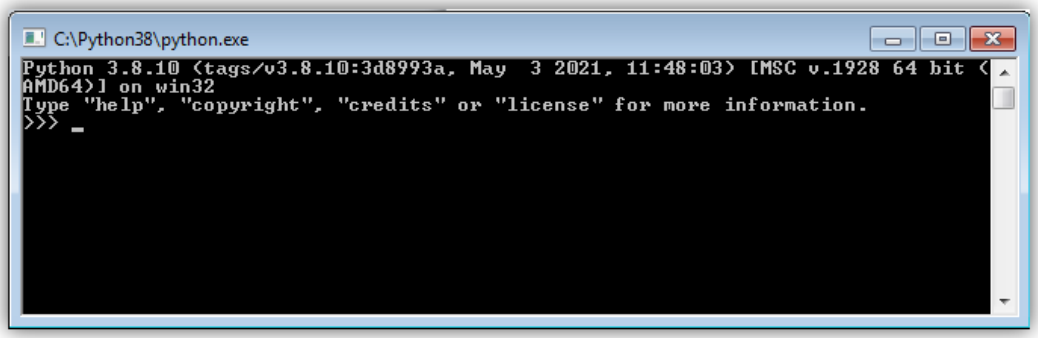

Рис. 1.6. Результат запуска интерпретатора Python в окне терминала

В этом окне введите программный код следующего содержания:

```
print("Hello, World!")
```

В результате вы увидите следующий ответ (рис. 1.7).

Получение такого результата означает, что установка интерпретатора Python прошла без ошибок.

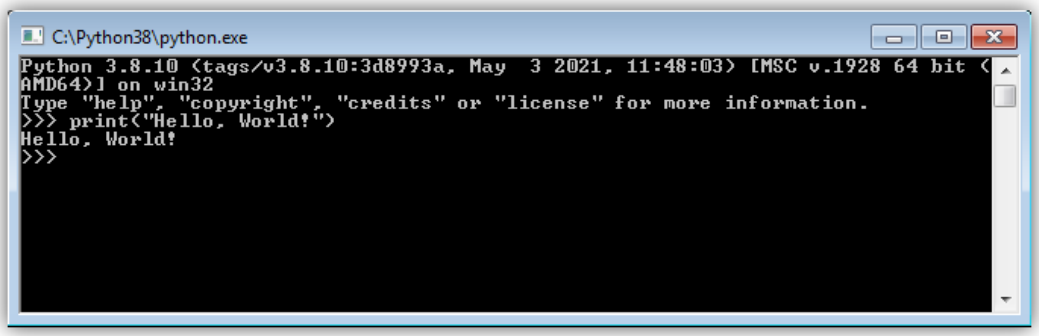

Рис. 1.7. Результат работы программы на Python в окне терминала

1.2. Интерактивная среда разработки программного кода PyCharm

В процессе разработки программных модулей удобнее работать в интерактивной среде разработки (IDE), а не в текстовом редакторе. Для Python одним из лучших вариантов считается IDE PyCharm от компании JetBrains. Для скачивания его дистрибутива перейдите по ссылке: **https://www.jetbrains.com/pycharm/download/** (рис. 1.8).

Рис. 1.8. Главное окно сайта для скачивания дистрибутива PyCharm

Эта среда разработки доступна для Windows, Linux и macOS. Существуют два вида лицензии PyCharm: **Professional** и **Community**. Мы будем использовать версию **Community**, поскольку она бесплатная и ее функционала более чем достаточно для наших задач. На момент подготовки этой книги была доступна версия PyCharm 2022.3.

1.2.1. Установка PyCharm в Windows

1. Запустите на выполнение скачанный дистрибутив PyCharm (рис. 1.9).
2. Выберите путь установки программы (рис. 1.10).

Рис. 1.9. Начальная заставка при инсталляции PyCharm

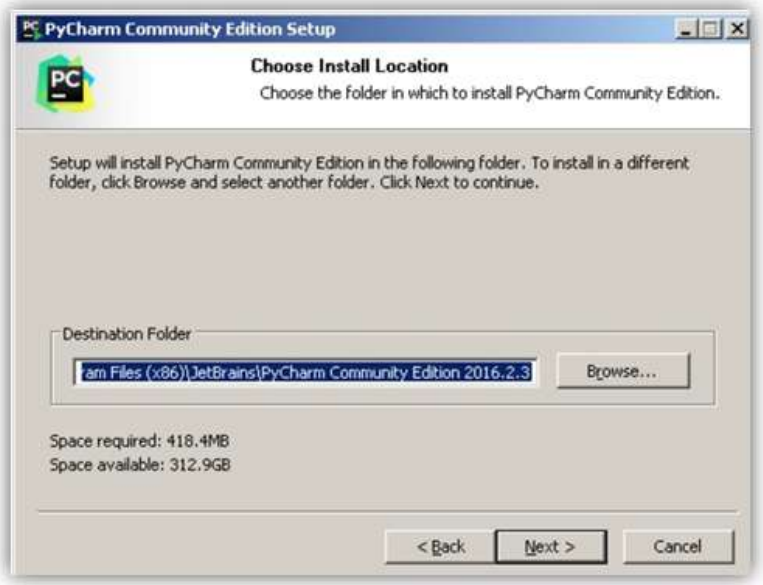

Рис. 1.10. Выбор пути установки PyCharm

3. Укажите ярлыки, которые нужно создать на рабочем столе (запуск 32- или 64-разрядной версии PyCharm), и отметьте флажком опцию **.py** в области **Create associations**, если требуется ассоциировать с PyCharm файлы с расширением py (рис. 1.11).
4. Выберите имя для папки в меню **Пуск** (рис. 1.12).
5. Далее PyCharm будет установлен на ваш компьютер (рис. 1.13).

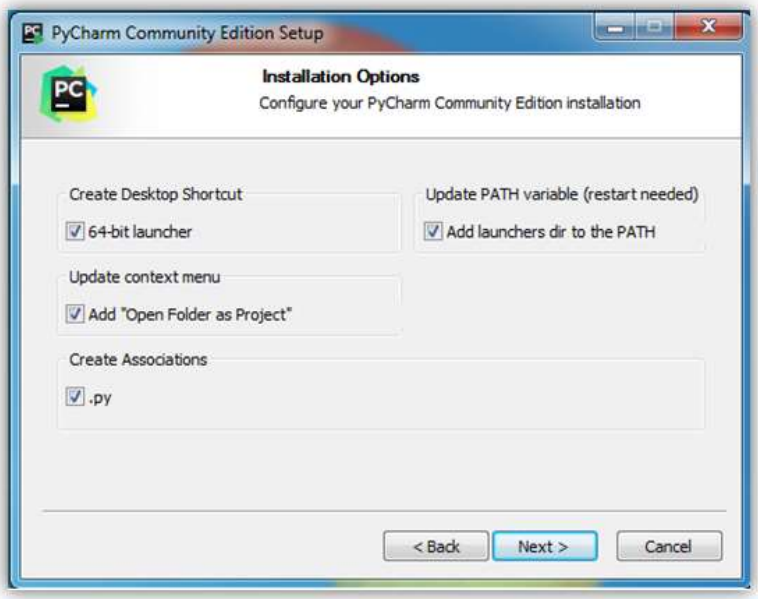

Рис. 1.11. Выбор разрядности устанавливаемой среды разработки PyCharm

Рис. 1.12. Выбор имени папки для PyCharm в меню **Пуск**

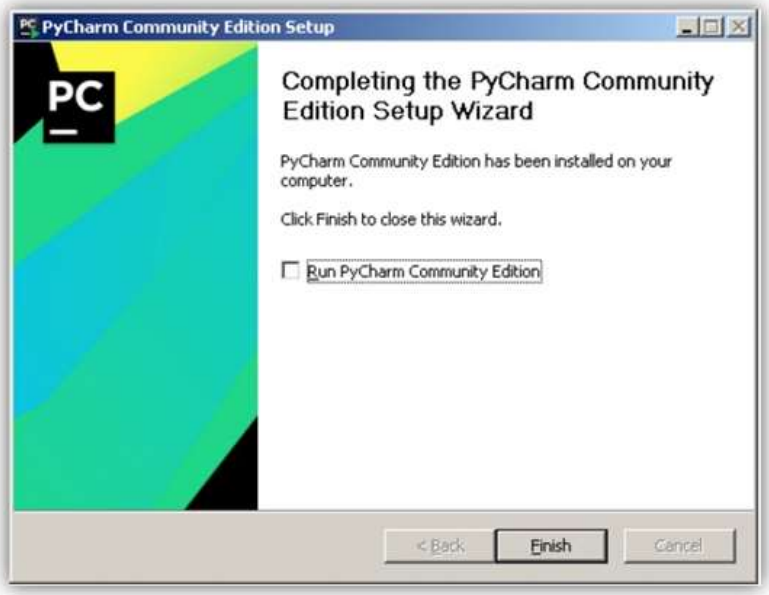

Рис. 1.13. Финальное окно установки пакета PyCharm

1.2.2. Установка PyCharm в Linux

1. Скачайте с сайта программы ее дистрибутив на свой компьютер.
2. Распакуйте архивный файл, для чего можно воспользоваться командой:

   ```
   > tar xvf имя_архива.tar.gz
   ```

   Результат работы этой команды представлен на рис. 1.14.

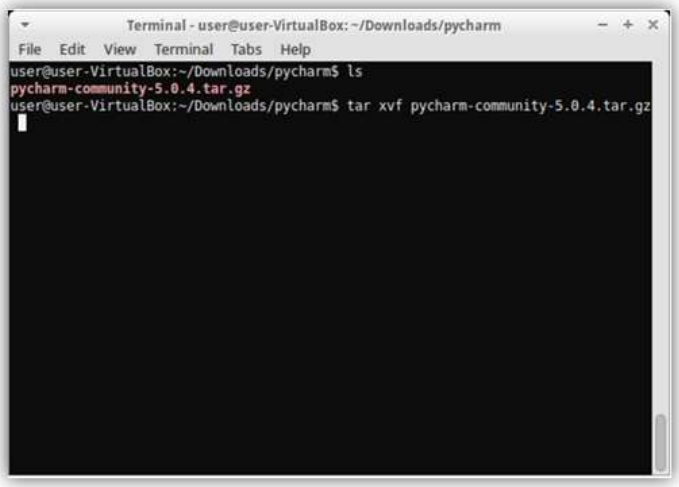

Рис. 1.14. Результат работы команды распаковки архива PyCharm

3. Перейдите в каталог, который был создан после распаковки дистрибутива, найдите в нем подкаталог bin и зайдите в него. Запустите установку PyCharm командой:

```
> ./pycharm.sh
```

Результат работы этой команды представлен на рис. 1.15.

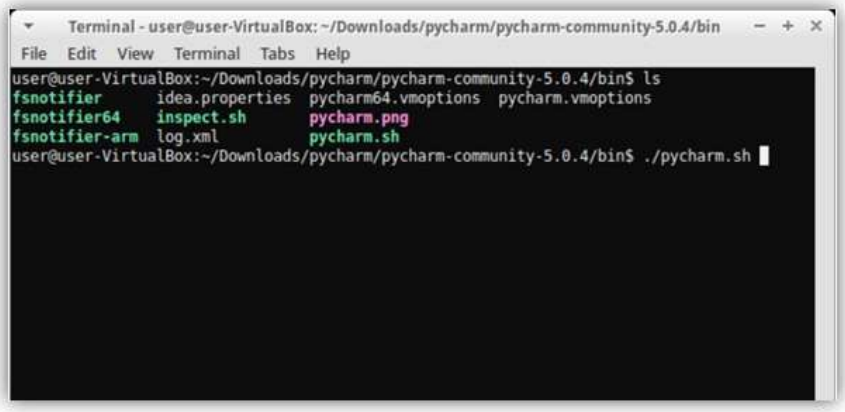

Рис. 1.15. Результаты работы команды инсталляции PyCharm

1.2.3. Проверка PyCharm

3. Запустите PyCharm и выберите вариант **Create New Project** в открывшемся окне (рис. 1.16).

Рис. 1.16. Создание нового проекта в среде разработки PyCharm

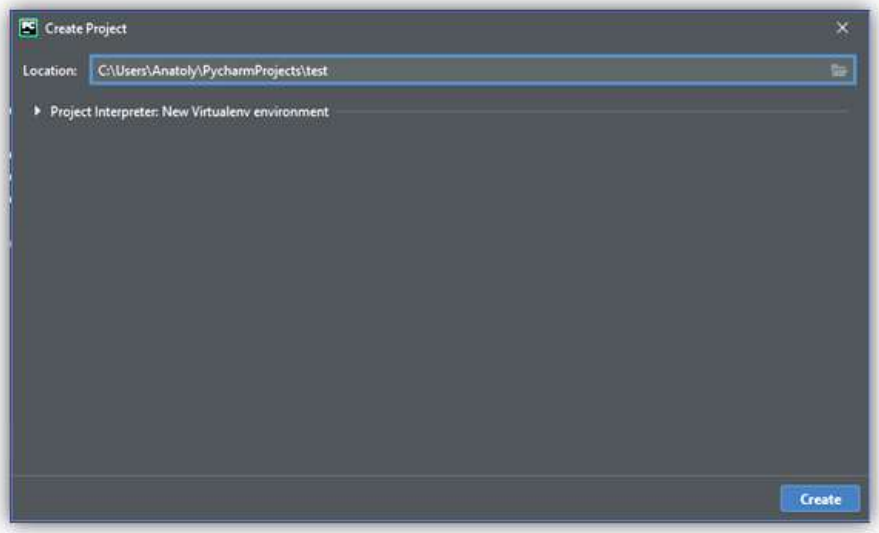

Рис. 1.17. Указание пути до проекта в среде разработки PyCharm

2. Укажите путь до создаваемого проекта Python и интерпретатор, который будет использоваться для его запуска и отладки (рис. 1.17).
3. Добавьте в проект файл, в котором будет храниться программный код Python (рис. 1.18).
4. Введите одну строчку кода программы (рис. 1.19).

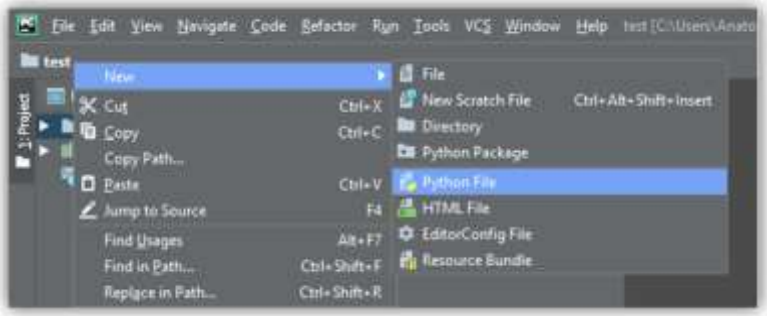

Рис. 1.18. Добавление в проект файла для программного кода на Python

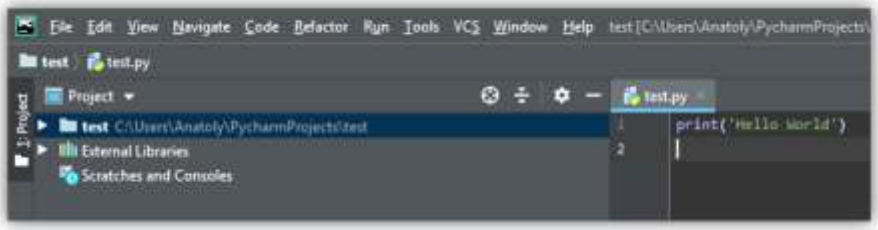

Рис. 1.19. Одна строка программного кода на Python в среде разработки PyCharm

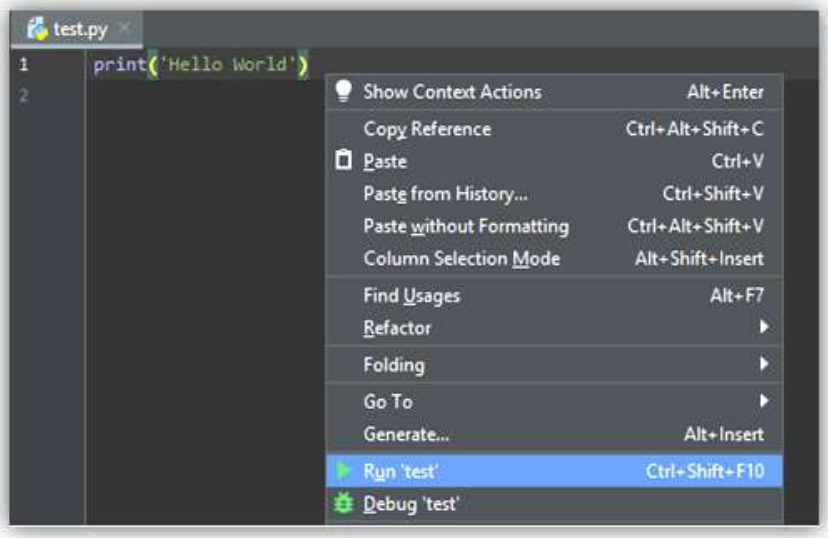

Рис. 1.20. Запуск строки программного кода на Python в среде разработки PyCharm

5. Запустите программу командой **Run** (рис. 1.20).

   В результате в нижней части экрана должно открыться окно с выводом результатов работы программы (рис. 1.21).

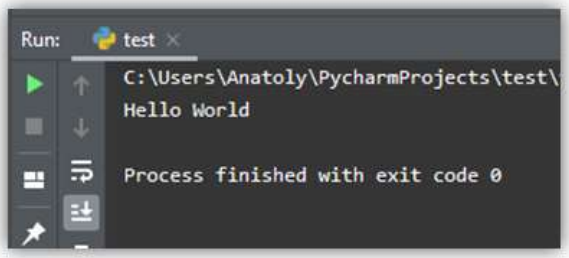

Рис. 1.21. Вывод результатов работы программы на Python в среде разработки PyCharm

1.3. Установка пакетов в Python с использованием менеджера пакетов pip

В процессе разработки программного обеспечения на Python часто возникает необходимость воспользоваться пакетом (библиотекой), который в текущий момент отсутствует на вашем компьютере.

В этом разделе вы узнаете о том, откуда можно взять нужный вам дополнительный инструментарий для разработки ваших программ. В частности:

- где взять отсутствующий пакет;
- как установить pip — менеджер пакетов в Python;
- как использовать pip;
- как установить пакет;
- как удалить пакет;
- как обновить пакет;
- как получить список установленных пакетов;
- как выполнить поиск пакета в репозитории.

1.3.1. Репозиторий пакетов программных средств PyPI

Необходимость в установке дополнительных пакетов возникнет довольно часто, поскольку решение практических задач обычно выходит за рамки базового функционала, который предоставляет Python. Это, например, создание веб-приложений, обработка изображений, распознавание объектов, нейронные сети и другие элементы искусственного интеллекта, геолокация и т. п. В таком случае необходимо узнать, какой пакет содержит функционал, который вам необходим, найти его, скачать, разместить в

нужном каталоге и начать использовать. Все указанные действия можно выполнить и вручную, однако этот процесс поддается автоматизации. К тому же скачивать пакеты с неизвестных сайтов может быть весьма опасно.

В рамках Python все эти задачи автоматизированы и решены. Существует так называемый Python Package Index (PyPI) — репозиторий, открытый для всех разработчиков на Python, в котором вы можете найти пакеты для решения практически любых задач. При этом у вас отпадает необходимость в разработке и отладке сложного программного кода — вы можете воспользоваться уже готовыми и проверенными решениями огромного сообщества программистов на Python. Вам нужно просто подключить требуемый пакет или библиотеку к своему проекту и активировать уже реализованный в них функционал. В этом и заключается преимущество Python перед другими языками программирования, когда небольшим количеством программного кода можно реализовать решение достаточно сложных практических задач. Там также есть возможность выкладывать свои пакеты. Для скачивания и установки нужных модулей в ваш проект существует специальная утилита, которая называется pip. Сама аббревиатура, которая на русском языке звучит как "пип", фактически раскрывается как "установщик пакетов" или "предпочитаемый установщик программ". Это утилита командной строки, которая позволяет устанавливать, переустанавливать и деинсталлировать PyPI пакеты простой командой `pip`.

1.3.2. pip — менеджер пакетов в Python

pip — утилита консольная (без графического интерфейса). После того, как вы ее скачаете и установите, она пропишется в PATH и будет доступна для использования. Эту утилиту можно запускать самостоятельно, например, через терминал Windows или в терминальном окне PyCharm командой:

```
> pip <аргументы>
```

pip можно запустить и через интерпретатор Python:

```
> python -m pip <аргументы>
```

Ключ `-m` означает, что мы хотим запустить модуль (в нашем случае `pip`).

При развертывании современной версии Python (начиная с Python 2.7.9 и более поздних версий) pip устанавливается автоматически. В PyCharm проверить наличие модуля pip достаточно просто — для этого нужно войти в настройки проекта через меню **File | Settings | Project Interpreter**. Модуль pip должен присутствовать в списке загруженных пакетов и библиотек (рис. 1.22).

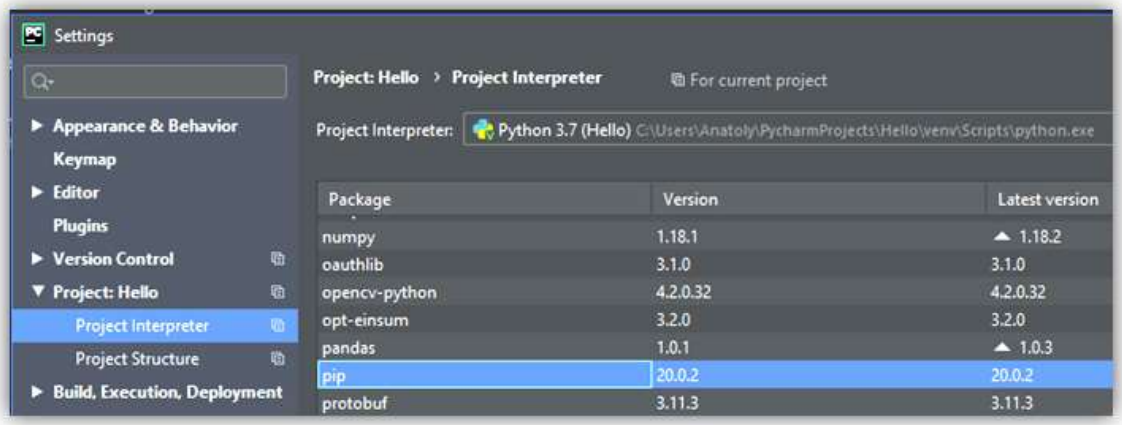

Рис. 1.22. Проверка наличия в проекте модуля pip

При отсутствии в списке этого модуля последнюю его версию можно загрузить, нажав на значок + в правой части окна и выбрав модуль pip из списка (рис. 1.23).

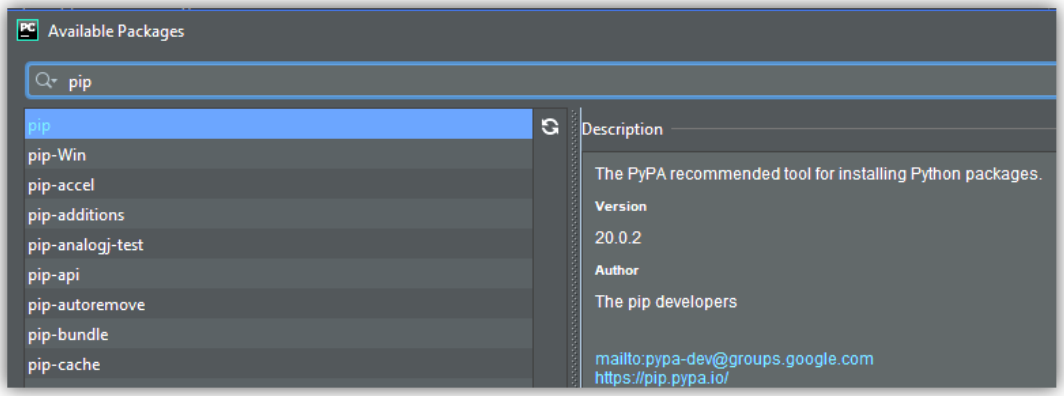

Рис. 1.23. Загрузка модуля pip

1.3.3. Использование менеджера пакетов pip

Здесь мы рассмотрим основные варианты использования pip: установку, удаление и обновление пакетов.

Pip позволяет установить самую последнюю версию пакета, конкретную версию или воспользоваться логическим выражением, через которое можно определить, что вам, например, нужна версия не ниже указанной. Также есть поддержка установки пакетов из репозитория. Рассмотрим эти варианты (здесь `Name` — это имя пакета).

- Установка последней версии пакета:

```
> pip install Name
```

- Установка определенной версии:

```
> pip install Name==3.2
```

- Установка пакета с версией не ниже 3.1:

```
> pip install Name>=3.1
```

- Для того чтобы удалить пакет, воспользуйтесь командой:

```
> pip uninstall Name
```

- Для обновления пакета задайте ключ `--upgrade`:

```
> pip install --upgrade Name
```

- Для вывода списка всех установленных пакетов служит команда:

```
> pip list
```

- Если вы хотите получить более подробную информацию о конкретном пакете, то укажите аргумент `show`:

```
> pip show Name
```

- Если вы не знаете точного названия пакета или хотите посмотреть на пакеты, содержащие конкретное слово, то вы можете это сделать, используя аргумент `search`:

```
> pip search "test".
```

Если вы запускаете pip в терминале Windows, то терминальное окно автоматически закроется после того, как эта утилита завершит свою работу. При этом вы просто не успеете увидеть результаты ее работы. Чтобы терминальное окно не закрывалось автоматически, команды pip нужно запускать в нем с ключом `/k`. Например, запуск процедуры установки пакета `tensorflow` должен выглядеть так, как показано на рис. 1.24.

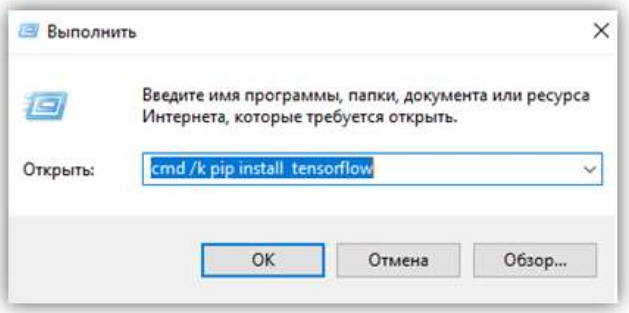

Рис. 1.24. Выполнение команды модуля pip в терминальном окне Windows

Если же пакет pip запускается из терминального окна PyCharm, то в указании дополнительных ключей нет необходимости, так как терминальное окно после завершения работы программ не закрывается. Пример выполнения той же команды в терминальном окне PyCharm показан на рис. 1.25.

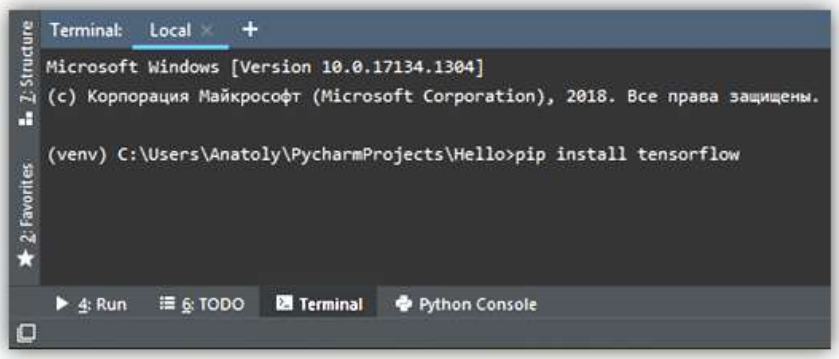

Рис. 1.25. Выполнение команды модуля pip в терминальном окне PyCharm

1.4. Фреймворк Django для разработки веб-приложений

Итак, основной инструментарий для разработки программ на языке Python установлен, и мы можем перейти к установке дополнительных библиотек. В этом разделе мы установим фреймворк Django, который позволяет разрабатывать веб-приложения.

Запустим среду разработки PyCharm и создадим в ней новый проект `Web_1`. Для этого в главном меню выберите опцию **File | New Project** (рис. 1.26).

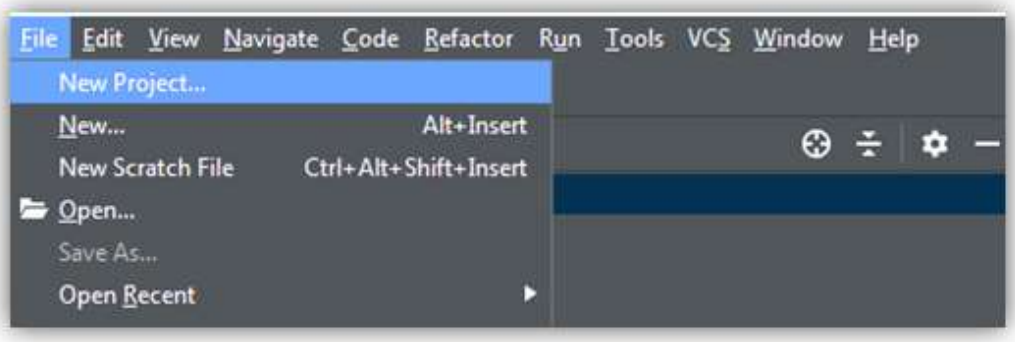

Рис. 1.26. Создание нового проекта в среде разработки PyCharm

Откроется окно, где вы можете задать имя создаваемому проекту, определить виртуальное окружение для этого проекта и указать каталог, в котором находится интерпретатор Python. Задайте новому проекту имя `Web_1` и нажмите кнопку **Create** (рис. 1.27).

Будет создан новый проект. Это, по сути дела, шаблон проекта, в котором пока еще ничего нет (рис. 1.28).

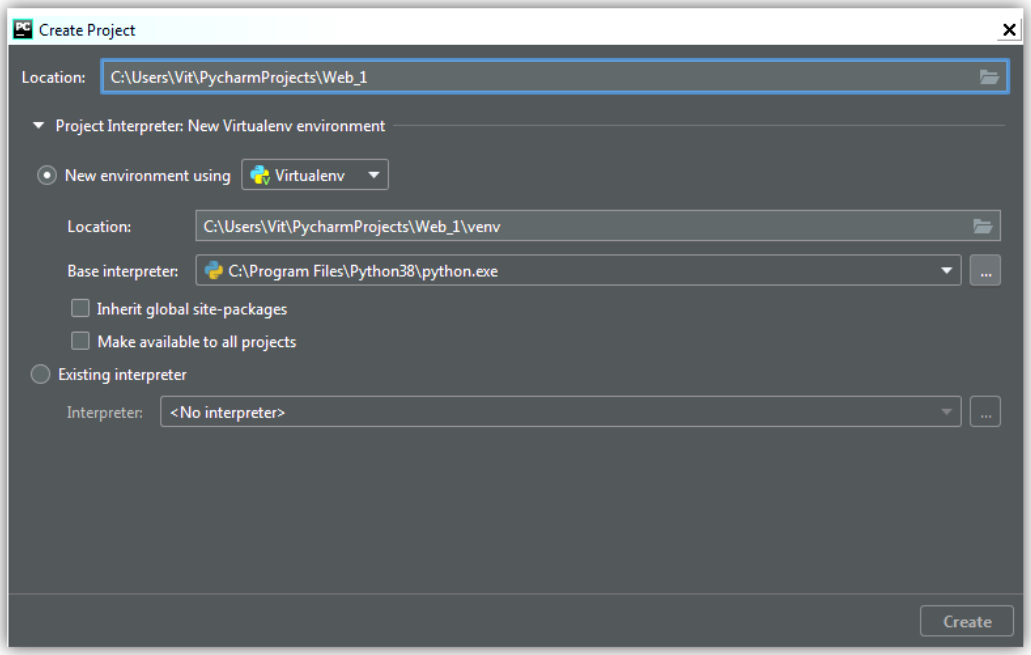

Рис. 1.27. Задаем новому проекту имя `Web_1` в среде разработки PyCharm

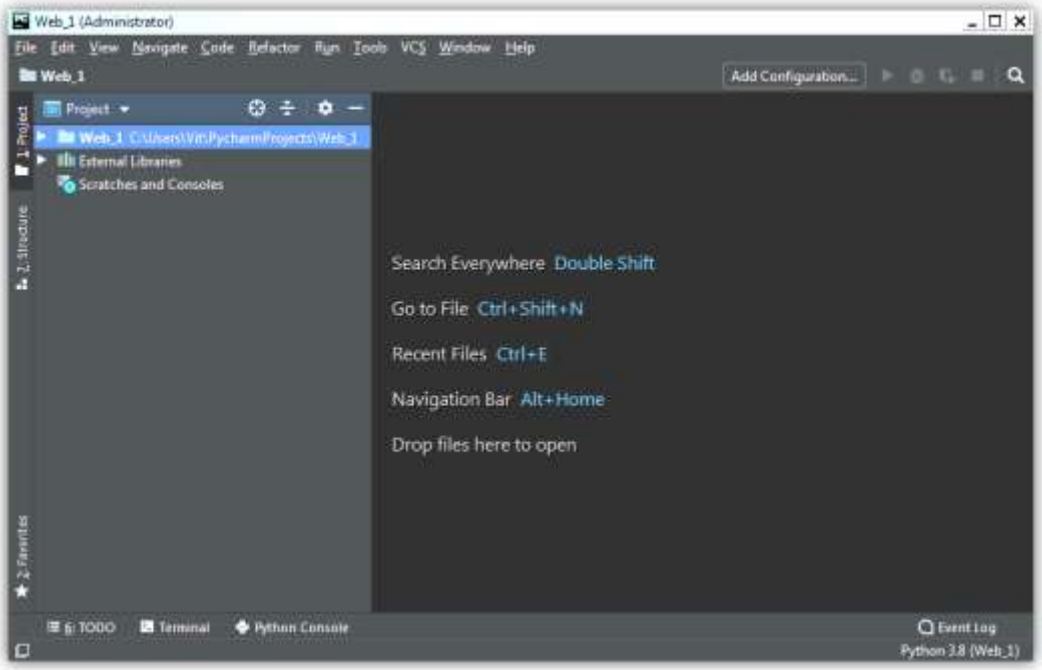

Рис. 1.28. Внешний вид интерфейса PyCharm с пустым проектом

Фреймворк Django — это фактически дополнительная библиотека к Python, и установить данный инструментарий можно так же, как и любую другую библиотеку. Для установки библиотеки Django можно в меню **File** выбрать опцию **Settings...** (рис. 1.29).

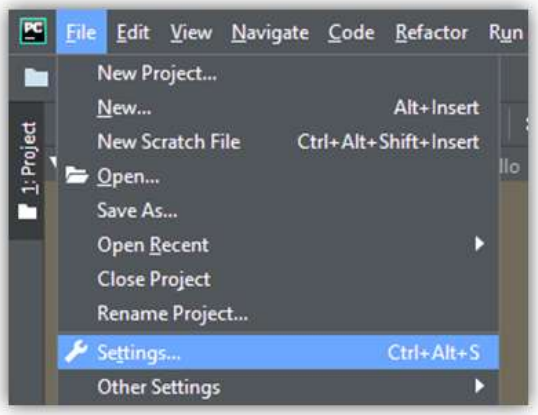

Рис. 1.29. Вызов окна **Settings** настройки параметров проекта

В левой части открывшегося окна настроек выберите опцию **Project Interpreter**, и в правой части окна будет показана информация об интерпретаторе языка Python и подключенных к нему библиотеках (рис. 1.30).

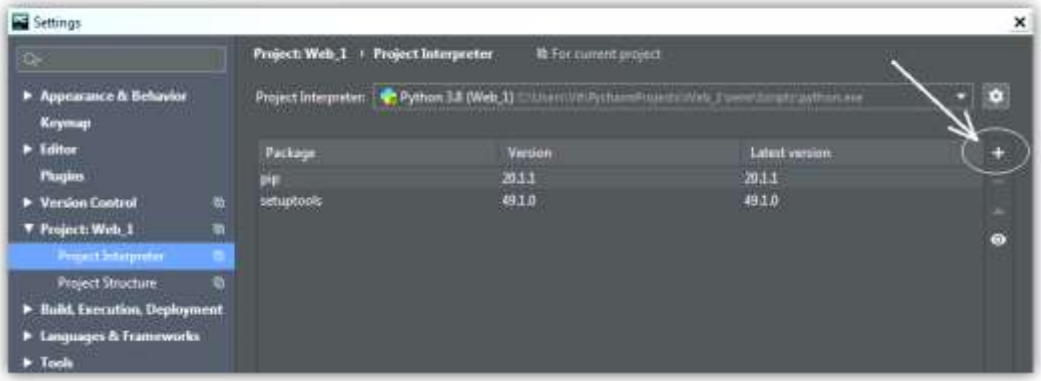

Рис. 1.30. Информация об интерпретаторе языка Python

Чтобы добавить новую библиотеку, нужно нажать на значок в правой части окна, после чего будет отображен полный список доступных библиотек. Здесь можно либо пролистать весь список и найти библиотеку Django, либо набрать наименование этой библиотеки в верхней строке поиска, и она будет найдена в списке (рис. 1.31).

Нажмите теперь на кнопку **Install Package**, и выбранная библиотека будет добавлена в ваш проект (рис. 1.32).

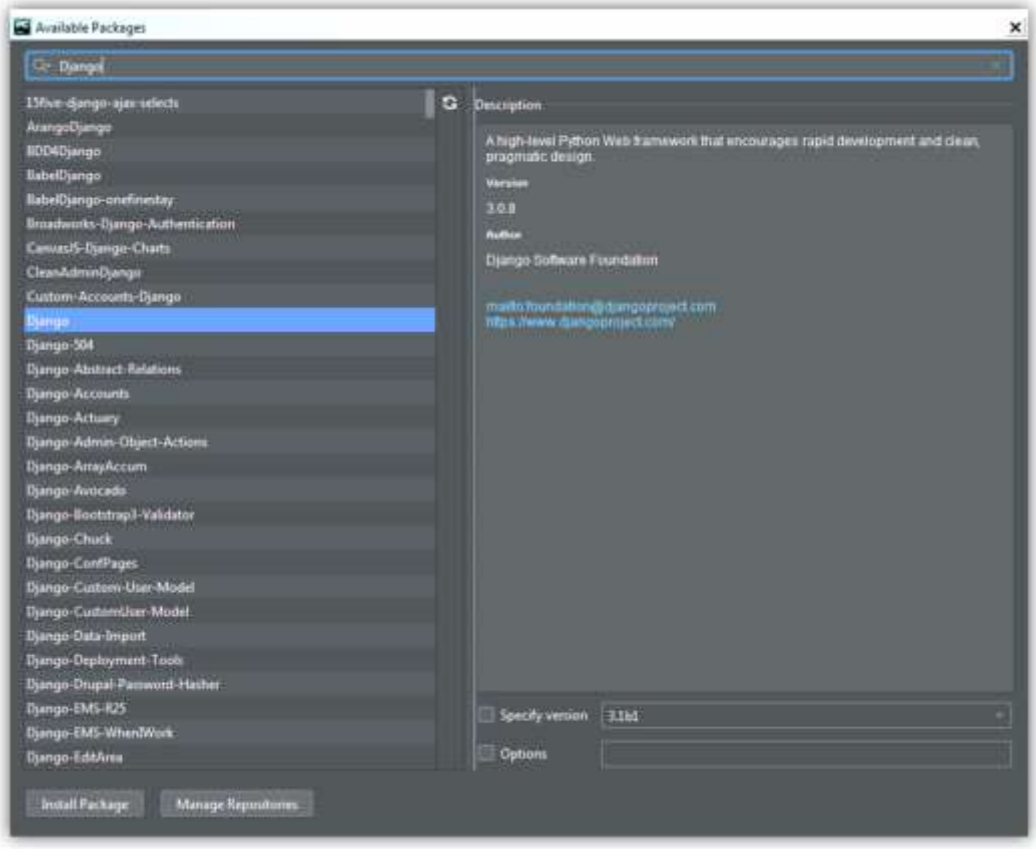

Рис. 1.31. Поиск библиотеки Django в списке доступных библиотек

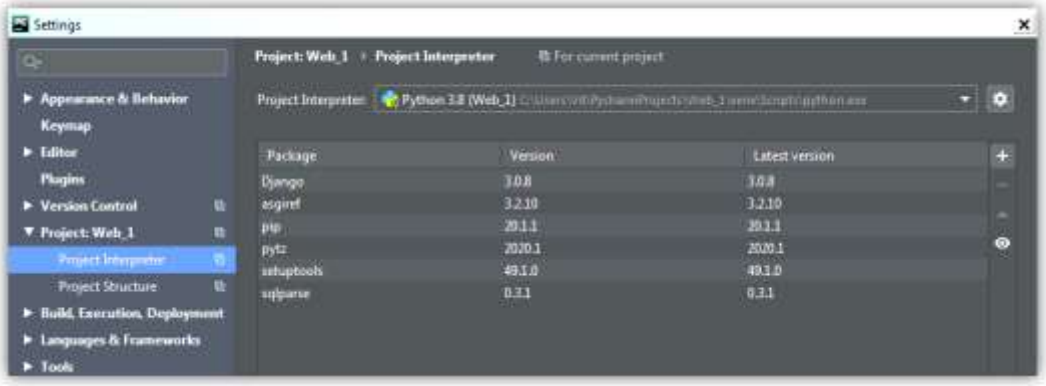

Рис. 1.32. Библиотека Django добавлена в список подключенных библиотек

На момент подготовки данной книги к изданию была доступен фреймворк Django версии 4.1.4. Теперь у нас есть минимальный набор инструментальных средств, который необходим для разработки веб-приложений на языке Python. Впрочем, в процессе рассмотрения конкретных примеров нам понадобится загрузка еще ряда дополнительных пакетов и библиотек. Их описание и процедуры подключения будут представлены в последующих главах.

1.5. Менеджер баз данных SQLiteStudio

Поскольку мы планируем создавать сайты, которые взаимодействуют с базами данных (БД), то для этого нам понадобится специальное программное средство. По умолчанию Django использует СУБД SQLite. Скачать установщик менеджера баз данных SQLite для ПК с ОС Windows можно по ссылке **https://sqlitestudio.pl/**.

После запуска загрузочного файла мы увидим, что это программное средство имеет русскоязычный интерфейс (рис. 1.33).

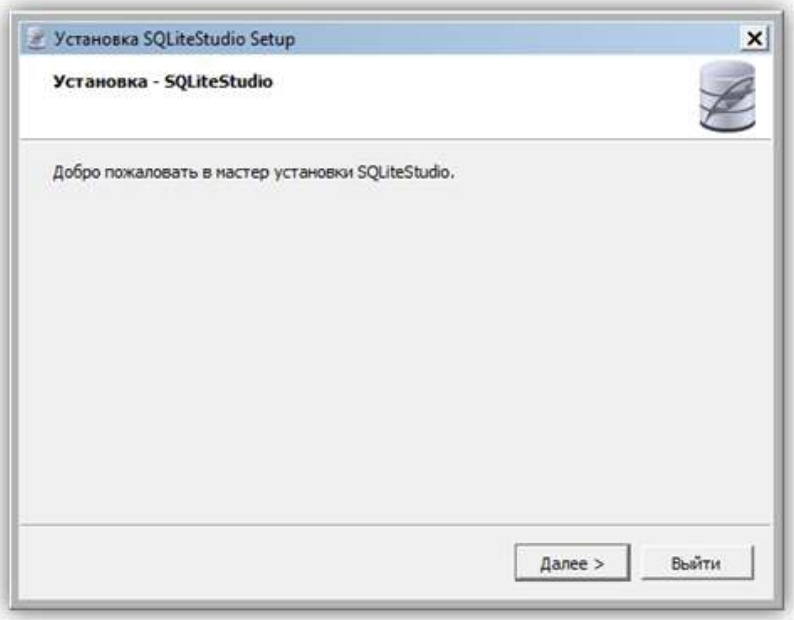

Рис. 1.33. Начальное окно установки менеджера SQLiteStudio

В следующих окнах, которые мы пропустим, следует выполнять рекомендации мастера установки. В завершающем окне нам будет предложено сразу запустить менеджер SQLiteStudio (рис. 1.34).

Нажмите здесь на кнопку **Завершить**, и откроется окно выбора языка интерфейса (рис. 1.35).

Выбираем русский язык и нажимаем на кнопку **OK**. Если процесс установки прошел корректно, то на экран будет выведено главное окно менеджера SQLiteStudio (рис. 1.36).

Как можно видеть, несмотря на выбор русского языка в окне выбора языка интерфейса (см. рис. 1.35), меню в главном окне менеджера SQLiteStudio выведено в англоязычном исполнении. Это, скорее всего, не совсем корректная локализация именно данной версии программного продукта — часть интерфейса осталась не локализованной.

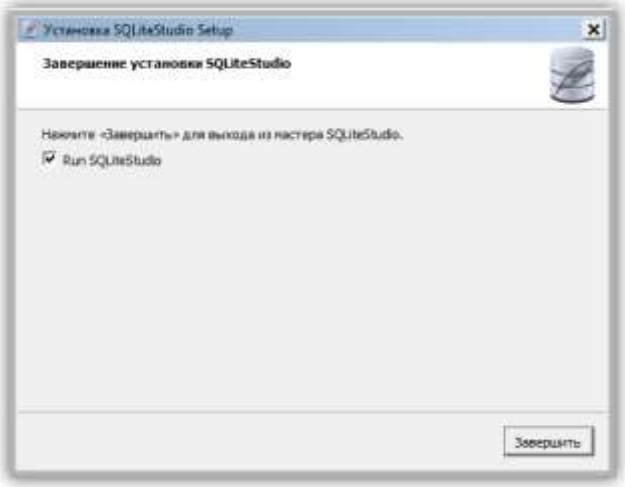

Рис. 1.34. Завершающее окно установки менеджера SQLiteStudio

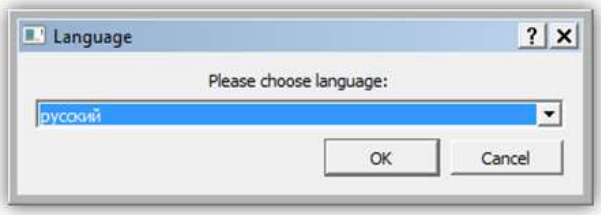

Рис. 1.35. Окно выбора языка интерфейса менеджера SQLiteStudio

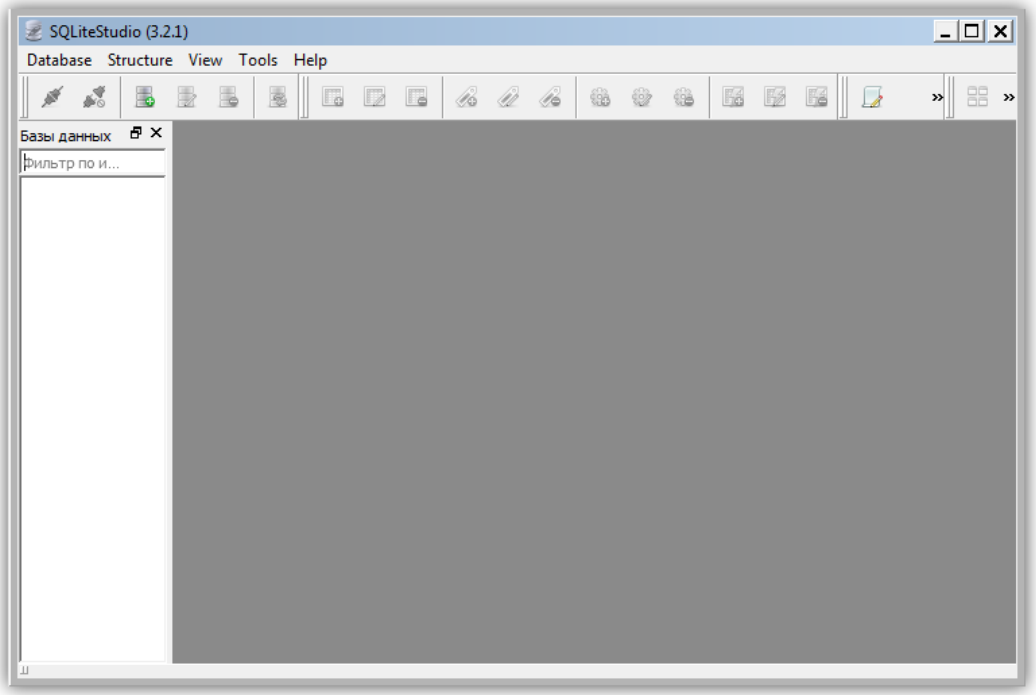

Рис. 1.36. Главное окно менеджера SQLiteStudio

1.6. Краткие итоги

В этой главе мы познакомились с основными инструментальными средствами, с помощью которых можно разрабатывать веб-приложения. Это интерпретатор Python, интерактивная среда разработки программного кода PyCharm, фреймворк Django для разработки веб-приложений и менеджер работы с базами данных SQLiteStudio. Установив на свой компьютер перечисленные инструментальные средства, теоретически можно приступать к написанию программного кода и созданию HTML-страниц. Мы пока не устанавливали фреймворк Bootstrap и сервер баз данных MySQL, поскольку этому инструментарию будут посвящены отдельные главы. Наличие инструмента — это необходимое, но не достаточное условие для того, чтобы приступить к программированию. Нужно еще и уметь применять этот инструментарий.

Те специалисты, которые имеют достаточный опыт программирования на Python, могут сразу перейти к следующей главе, в которой дается некоторое представление о веб-технологиях и языке разметки HTML. Для тех же, кто только начинает знакомиться с миром Python, нужно получить хотя бы элементарные навыки работы с этим языком программирования, обратившись к предназначенной для этих целей специальной литературе.

Итак, переходим к следующей главе и знакомимся с технологиями веб-программирования.

Глава 2. Веб-технологии и базовые сведения об HTML

В настоящее время веб-технологии стремительно развиваются, проникая в самые разнообразные сферы нашей жизни. Для компаний присутствие в сети Интернет — это возможность рассказать о своих товарах и услугах, найти потенциальных партнеров и клиентов, снизить издержки за счет интернет торговли и "облачных" сервисов. Рядовые пользователи уже привыкли к услугам интернет-магазинов, интернет-банкинга, общаются в социальных сетях, могут получать через Интернет государственные услуги.

Из материалов этой главы вы также узнаете:

- что такое веб-технологии;
- в чем различия технологий клиентского и серверного программирования;
- какие фреймворки предназначены для создания веб-приложений на Python;
- какие теги помогут представить текст на HTML-страницах;
- что такое каскадные таблицы стилей;
- какие возможности таит в себе скриптовый язык JavaScript.

> ***Примечание.***
> Программный код листингов данной главы вы можете запустить в интерфейсе интерактивной цифровой книги. Для этого в левой панели главного окна нужно открыть список листингов главы 2 и активировать ссылку на тот или иной листинг. После того, как откроется страница с программным кодом, необходимо левой кнопкой мыши нажать на кнопку «**Выполнить код**»

2.1. Базовые сведения о веб-технологиях

Под веб-технологиями в дальнейшем мы будем понимать всю совокупность средств для организации взаимодействия пользователей с удаленными приложениями. Поскольку в каждом сеансе взаимодействуют две стороны (сервер и клиент), то и веб-приложения можно разделить на две группы: приложения на стороне сервера (server-side) и приложения на стороне клиента (client-side). Благодаря веб-приложениям удаленному пользователю доступны не только статические документы, но и сведения из базы данных. Использование баз данных в Интернете приобрело огромную популярность и практически стало отдельной отраслью компьютерной технологии.

Но для начала разберемся с основными понятиями веб-технологий: что такое веб-сайт и веб-страница. Часто неопытные пользователи неправомерно смешивают эти понятия. *Веб-страница* — это минимальная логическая единица в сети Интернет, которая представляет собой документ, однозначно идентифицируемый уникальным интернет-адресом (URL, Uniform Resource Locator — унифицированный указатель ресурса). *Веб-сайт* — это набор тематически связанных веб-страниц, находящихся на одном сервере и принадлежащих одному владельцу. В частном случае веб-сайт может состоять из единственной веб-страницы. Сеть Интернет является совокупностью всех веб-сайтов.

Для формирования веб-страниц используется *язык разметки гипертекста* HTML (Hyper Text Markup Language). С его помощью осуществляется логическая (смысловая) разметка документа (веб-страницы). Для управления внешним видом веб-страниц применяются *каскадные таблицы стилей* (от англ. Cascading Style Sheets, CSS). Сформированные на сервере HTML-документы можно просмотреть на компьютере клиента с помощью специальной программы — *браузера*.

Как было отмечено ранее, совокупность связанных между собой веб-страниц представляет собой веб-сайт. Сайты можно условно разделить на статические и динамические.

Статический сайт — это сайт с неизменным информационным наполнением. Основу статического сайта составляют веб-страницы с постоянным содержимым, разработанные на основе стандартной HTML-технологии. Страницы сайта хранятся в виде HTML-кода в файловой системе сервера. Естественно, на таком сайте могут присутствовать различные видеоролики и анимация. Основная отличительная особенность статического сайта заключается в том, что веб-страницы такого сайта создаются заранее. Для редактирования содержимого страниц и обновления сайта страницы модифицируют вручную с применением HTML-редактора и затем заново загружают на сайт. Схема взаимодействия клиента со статическим сайтом приведена на рис. 2.1.

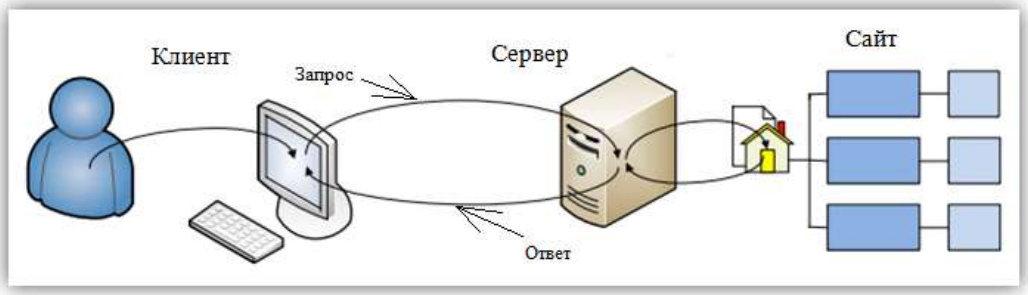

Рис. 2.1. Схема взаимодействия клиента со статическим сайтом

Такая схема приемлема в том случае, когда содержимое сайта довольно постоянно и изменяется сравнительно редко. Если же информация, размещенная на статическом сайте, требует регулярной актуализации и обновления, то для его поддержания неизбежны внушительные трудозатраты. Таким образом, статический сайт дешевле в разработке и технической поддержке, но эти достоинства могут нивелироваться серьезными недостатками, связанными с необходимостью оперативного обновления актуальной информации и достаточно высокой трудоемкостью модификации.

Динамический сайт — это сайт с динамическим информационным наполнением. Динамические страницы также формируются с помощью HTML, но такие страницы обновляются постоянно, нередко при каждом новом обращении к ним. Динамические сайты основываются на статических данных и HTML-разметке, но дополнительно содержат программную часть (скрипты), а также базу данных (БД), благодаря которым страница "собирается" из отдельных фрагментов в режиме реального времени. Это позволяет обеспечить гибкость в подборе и представлении информации, соответствующей конкретным запросам посетителей сайта.

Таким образом, динамический сайт состоит из набора различных блоков: шаблонов страниц, информационного наполнения (контента) и скриптов, хранящихся в виде отдельных файлов. Иными словами, динамическая веб-страница формируется из страницы-шаблона и добавляемых в шаблон динамических данных. Тип данных, загружаемых в шаблон, будет зависеть от того, какой запрос сделал клиент. Схема взаимодействия клиента с динамическим сайтом приведена на рис. 2.2.

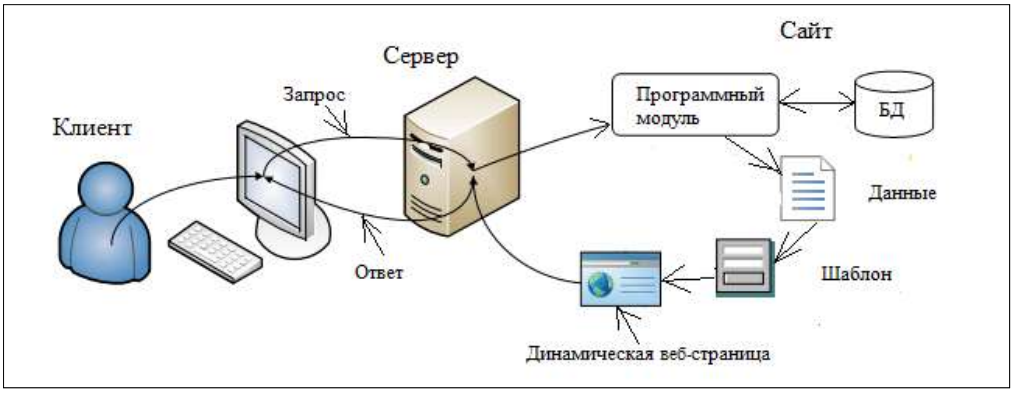

Рис. 2.2. Схема взаимодействия клиента с динамическим сайтом

Динамичность сайта заключается в том, что для изменения страницы достаточно поменять ее информационное наполнение, а сам механизм формирования и вывода страницы остается тем же. Кроме получения различных данных, удаленный клиент может изменить и содержание информационной части сайта. Для этого используются *веб-формы*. Клиент заполняет веб-форму своими данными, и эта информация вносится в БД сайта.

Динамические сайты различаются в зависимости от применяемых технологий и процесса получения динамических страниц. Динамические страницы можно получить следующими способами:

- генерацией страницы на стороне сервера, осуществляемой серверными скриптами на языках PHP, Perl, ASP.NET, Java, Python и др. При этом информационное наполнение страниц хранится в базах данных;
- генерацией страницы на стороне клиента (JavaScript);
- комбинированной генерацией. Чаще всего на практике встречается именно комбинация первых двух способов.

2.1.1. Технологии клиентского программирования

Простейшим средством "оживления" веб-страниц, добавления динамических эффектов и задания реакции на пользовательские действия является скриптовый язык программирования JavaScript. Сценарии (скрипты) JavaScript внедряются непосредственно в веб-страницу или связываются с ней и после загрузки страницы с сервера выполняются браузером на стороне клиента. Все современные браузеры имеют поддержку JavaScript.

JavaScript — это компактный объектно-ориентированный язык для создания клиентских веб-приложений. Этот язык применяется для обработки событий, связанных с вводом и просмотром информации на веб-страницах. JavaScript обычно используется в клиентской части веб-приложений, в которых клиентом выступает браузер, а сервером — удаленный веб-сервер. Обмен информацией в веб-приложениях происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения технологии клиентского программирования представляют собой кросс-платформенные сервисы. Язык сценариев JavaScript позволяет создавать интерактивные веб-страницы и содержит средства управления окнами браузера, элементами HTML-документов и стилями CSS.

2.1.2. Технологии серверного программирования

Без серверного программирования нельзя обойтись, если необходимо изменять и сохранять какую-либо информацию, хранящуюся на сервере (например, организовать прием и сохранение сообщений, отправляемых пользователями). Без серверных скриптов невозможно представить себе гостевые книги, форумы, чаты, опросы (голосования), счетчики посещений и другие программные компоненты, которые активно взаимодействуют с базами данных. Серверное программирование позволяет решать такие задачи, как регистрация и авторизация пользователей, управление аккаунтом (в почтовых веб-системах, социальных сетях и др.), поиск информации в базе данных, работа интернет-магазина и т. п. Серверные языки программирования открывают перед программистом большие функциональные возможности. Современные сайты зачастую представляют собой чрезвычайно сложные программно-информационные системы, решающие разнообразные задачи, и теоретически могут дублировать функции большинства бизнес-приложений.

Работа серверных скриптов зависит от платформы, т. е. от того, какие технологии поддерживаются сервером, на котором расположен сайт. Например, основной технологией, поддерживаемой компанией Microsoft, является ASP.NET (Active Server Pages, активные серверные страницы). Другие серверы могут поддерживать языки Perl, PHP, Python.

- **Perl** — высокоуровневый интерпретируемый динамический язык программирования общего назначения. Он является одним из наиболее старых языков, используемых для написания серверных скриптов. По популярности Perl сейчас уступает более простому в освоении языку PHP. В настоящее время используются бесплатные технологии EmbPerl и mod_perl, позволяющие обрабатывать HTML-страницы со вставленными скриптами на языке Perl.
- **PHP** (Personal Home Page) — язык сценариев общего назначения, в настоящее время интенсивно применяемый для разработки веб-приложений. PHP-код может внедряться непосредственно в HTML. Это язык с открытым кодом, крайне простой для освоения, но вместе с тем способный удовлетворить запросы профессиональных веб-программистов, т. к. имеет большой набор специализированных встроенных средств.
- **Python** — универсальный язык программирования, применимый, в том числе и для разработки веб-приложений. Важно, что в Python приложение постоянно находится в памяти, обрабатывая множество запросов пользователей без "перезагрузки". Таким образом, поддерживается правильное предсказуемое состояние приложения. В зависимости от того, какой фреймворк будет использоваться совместно с Python, взаимодействия могут существенно упрощаться. Например, Django, который сам написан на Python, имеет систему шаблонов для написания специальных HTML-файлов, которые могут включать код Python и взаимодействовать с данными из СУБД.

2.1.3. Фреймворки Django и Bootstrap для разработки веб-приложений

Любое веб-приложение состоит из клиентской части и серверной части, которые тесно связаны между собой (рис. 2.3).

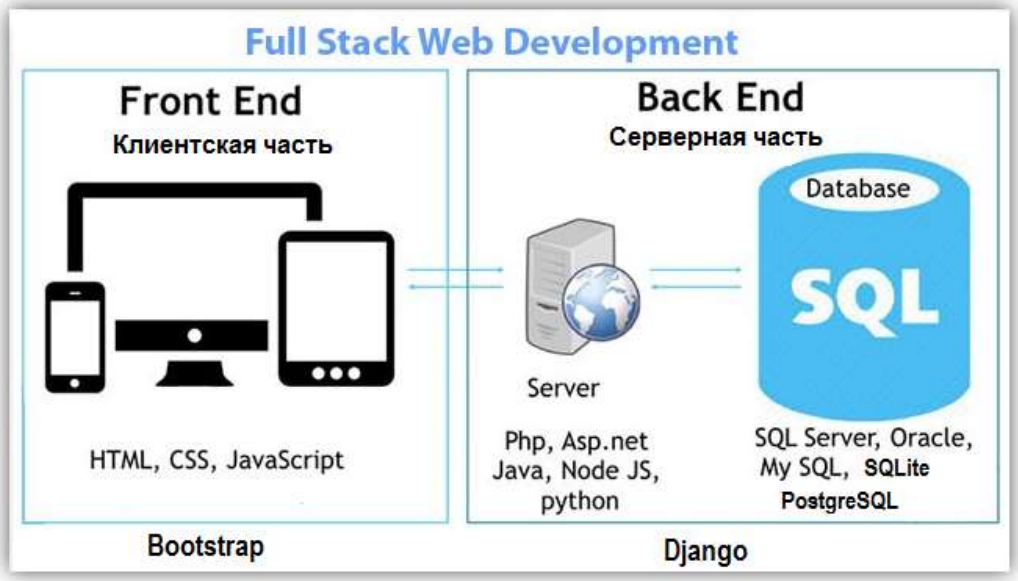

Рис. 2.3. Структура веб-приложения с фреймворками Django и Bootstrap

Каждая из этих частей разрабатывается с использованием разного набора инструментов, соответственно и к программистам, разрабатывающим ту или иную часть приложения, предъявляются различные требования. Учитывая эти особенности, различают три типа специалистов, занимающихся разработкой веб-приложений:

- ❒ разработчики клиентской части приложения (frontend-разработчики);
- ❒ разработчики серверной части приложения (backend-разработчики);
- ❒ разработчики полного стека (full stack web development).

Остановимся на различиях этих направлений, их особенностях и применяемых инструментальных средствах.

Понятие frontend-разработки. Под понятием frontend подразумевается разработка видимого для пользователя интерфейса и всех функций, с которыми он может взаимодействовать. По сути, когда вы переходите на любой сайт, то видите там кнопки, текст, различные изображения, анимацию и другие составляющие — все это реализовано при помощи инструментария frontend. Эти элементы создают на трех разных языках: HTML, CSS и JavaScript.

Основной инструмент в этой сфере — язык гипертекстовой разметки HTML. Он нужен в основном для разметки документа, т. е. HTML-страниц. С помощью него разработчик создает структуру, добавляет заголовки, списки и форматирует контент.

В связке с языком HTML используется и язык CSS (Cascading Style Sheets), который отвечает за внешний вид страницы. С его помощью оформляются цвета, шрифты и фоны различных блоков. Если выразиться простыми словами, то CSS служит для красивого оформления страницы и настройки ее внешнего вида уже после того, как основная структура была написана при помощи HTML.

С помощью JavaScript (JS) реализуется выполнение различных действий на странице, т. е. добавляется анимация и отклик на запросы пользователя. Например, страница реагирует на перемещение курсора и нажатие на кнопки мыши, изменяя поведение элементов в соответствии с действиями пользователя. Благодаря JS осуществляются действия на стороне клиента без необходимости перезагрузки страницы.

Backend-разработка. Под понятием Backend подразумевается разработка бизнес-логики веб-приложения. Эта часть отвечает за взаимодействие пользователя с данными, которые потом отображает клиентская часть приложения. Попросту говоря, это то, что скрыто от глаз пользователя и происходит на удаленном сервере вне его браузера и компьютера. Как только поступает запрос от пользователя (например, когда к элементу интерфейса подведен курсор и нажата кнопка мыши), сигнал сразу же отправляется на сервер, где и обрабатывается для дальнейшего вывода информации на экран. Это и есть логика сайта, заключающаяся в трех простых шагах:

6. Получение запроса от пользователя на сервер.
7. Обработка запроса пользователя на сервере.
8. Отправка ответа пользователю в его браузер.

При создании клиентской части приложения разработчики всегда используют описанные ранее языки программирования: HTML, CSS и JavaScript. Как правило, эти языки объединяются в рамках одного инструментального средства, среди которых, пожалуй, наиболее популярен фреймворк Bootstrap, тесно интегрированный с Django. С его помощью мы и будем разрабатывать клиентскую часть веб-приложений.

С разработкой серверной части дело обстоит немного иначе. Выбор языка и инструментария зависит от сервера. Но, как правило, разработчики выбирают один из универсальных языков программирования: Java, PHP, Python, Ruby и др.

В данной книге мы будем использовать язык программирования Python и следующие фреймворки:

- Django для разработки серверной части веб-приложений;
- Bootstrap для разработки клиентской части веб-приложений.

Что касается Django, то это инструмент, позволяющий упростить реализацию взаимодействия пользователей с базами данных, логику обработки данных на сервере и формирование динамических веб-страниц на основе шаблонов. Кроме того, административная панель Django дает возможность владельцам сайтов осуществлять удаленное администрирование развернутых сайтов, их модификацию, управление доступом пользователей и техническую поддержку. Этот фреймворк предназначен для backend-разработчиков, соответственно с его помощью повышается эффективность разработки серверной части приложения, но в паре с ним необходимы другие инструменты для создания привлекательных HTML-страниц.

На сегодняшний день наиболее популярный инструмент для frontend-разработчиков — фреймворк Bootstrap, поскольку он позволяет создавать адаптивные веб-приложения. Адаптивность заключается в том, что интерфейс приложения автоматически адаптируется под разрешение экрана того устройства, на котором оно было запущено, а значит приложение выглядит одинаково хорошо на экране смартфона, планшета и настольного компьютера. Фреймворк Bootstrap использует языки программирования HTML, CSS и JavaScript, на основе которых можно создавать как структуру HTML-страниц, так и красочное их оформление. Это свободно распространяемый инструментарий, который хорошо интегрируется с Django.

Прежде чем переходить к изучению Django и Bootstrap, рассмотрим базовые сведения об HTML, CSS, JavaScript.

2.2. Базовые сведения о HTML

Как было отмечено ранее, язык разметки гипертекста HTML является основой для формирования веб-страниц. С помощью HTML осуществляется логическое форматирование документа, и он предназначен только для этих целей.

HTML-документы строятся на основе тегов, которые структурируют документ. Обычно теги бывают парными, т. е. состоят из открывающего и закрывающего тега, хотя бывают и исключения. Имена открывающих тегов заключаются в угловые скобки `< ... >`, а закрывающие теги обрамляются угловыми скобками и знаком слеш `</ ... >`.

Весь HTML-документ обрамляется парными тегами `<html>...</html>`. Кроме того, для обеспечения корректного отображения документа современный стандарт требует наличия одиночного тега `<!DOCTYPE>`, который может иметь следующую структуру:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

В самом простом случае этот одиночный тег выглядит так:

```
<!DOCTYPE html>
```

Сами HTML-документы состоят из заголовка и тела. Заголовок документа описывается парой тегов `<head>...</head>`, а тело документа обрамляется парными тегами `<body>...</body>`. Таким образом, каркас HTML-документа будет иметь следующую структуру (листинг 2.1).

Листинг 2.1. Структура HTML-документа

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.1.

Заголовок может включать в себя несколько специализированных тегов, основными из которых являются `<title>...</title>` и `<meta>...</meta>`.

Тег `<title>` содержит информацию, которая будет выводиться в заголовочной части окна браузера пользователя. Например, если в этом теге имеется текст

```
<title>Мир книг</title>
```

то он отобразится в окне браузера так, как показано на рис. 2.4.

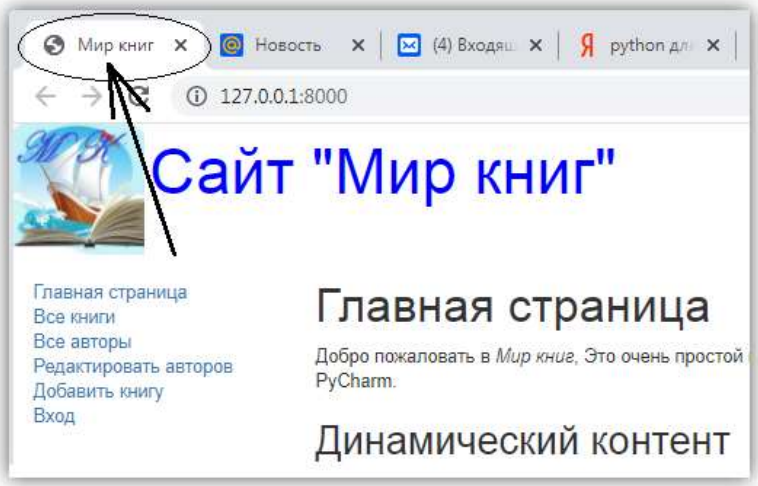

Рис. 2.4. Вывод содержимого тега `<title>` в браузере пользователя

Тег `<meta>` содержит специальную информацию:

- тип кодировки:

```
<meta charset="utf-8" />
```

- список ключевых слов:

```
<meta name="keywords" content="СУБД, БД, Базы данных">
```

В первом случае этот тег обеспечивает поддержку необходимой кодировки, а во втором — позволяет поисковым машинам корректно индексировать страницы сайта по ключевым словам. Следует заметить, что современные поисковые системы могут игнорировать ключевые слова, но это не отменяет возможность использования данного атрибута.

В рассмотренных тегах фрагменты `name="keywords"` и `content="`*`список ключевых слов`*`"` представляют собой *атрибуты* (параметры) тегов, которые конкретизируют их. Например, атрибуты могут указывать, что текст, заключенный в том или ином теге, при отображении должен выравниваться по центру. Атрибуты записываются сразу после имени тега, причем значения атрибутов заключаются в кавычки. Атрибутов у тега может быть несколько, но могут они и вовсе отсутствовать.

Приведем пример простейшей веб-страницы (листинг 2.2).

Листинг 2.2. Простейшая веб-страница

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.2.

Текст страницы можно набрать в любом редакторе и сохранить в файле с расширением htm или html. Такой файл можно открыть при помощи какого-либо браузера.

2.2.1. Теги для представления текста на HTML-страницах

Текст, содержащийся в теле документа, обычно обрамляется специальными тегами. Наиболее распространены следующие виды тегов.

Тег `<hi>` — для вывода заголовков, где `i` имеет значения от 1 до 6. Например:

```
<h1> ... </h1>
<h2> ... </h2>
<h3> ... </h3>
<h4> ... </h4>
<h5> ... </h5>
<h6> ... </h6>
```

Заголовки отображаются полужирным шрифтом. При этом у текста заголовка в теге <h1> наибольший размер шрифта, а у тега <h6> — наименьший. К этому тегу может быть добавлен параметр align, который определяет правило выравнивания заголовка относительно одной из сторон документа. Этот параметр может принимать следующие значения:

- left — выравнивание по левому краю;
- center — по центру;
- right — по правому краю;
- justify — выравнивание по ширине.

Тег <P> — для вывода параграфов (абзацев). К этому тегу также может быть добавлен параметр align. Каждый новый тег <P> с этим параметром устанавливает выравнивание параграфа относительно одной из сторон документа.

**Тег
** — служит для перевода строки.

Тег <HR> — предназначен для вывода горизонтальной линии. Параметр align этого тега определяет выравнивание линии относительно одной из сторон документа. Этот тег имеет еще несколько параметров:

- size — высота линии;
- width — ширина линии;
- color — цвет линии;
- noshade — отключает отбрасывание тени.

Представленные далее теги меняют следующие параметры текста:

- <B> — полужирный текст;
- <I> — наклонный текст;
- <U> — подчеркнутый текст;
- <TT> — моноширинный текст.

Тег <PRE> — выводит заранее отформатированный текст. Используется он для того, чтобы текст с пробелами, символами перехода на новые строки и символами табуляции корректно отображался браузером. В секциях <PRE> могут присутствовать гипертекстовые ссылки, однако применять другие HTML-теги нельзя.

Тег <FONT> — задает некоторые свойства шрифта. Эти свойства определяются следующими параметрами:

- size — размер шрифта от 1 до 7;
- face — начертание шрифта;
- color — цвет шрифта.

В листинге 2.3 приведен пример использования некоторых тегов.

Листинг 2.3. Пример использования тегов

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.3.

Открывающие теги могут содержать дополнительную информацию в виде параметров (атрибутов), которые существенно расширяют возможности представления информации. Параметры в открывающем теге записываются после названия тега в виде параметр="*значение*" и разделяются пробелами. Порядок следования параметров в теге не произвольный. Если параметр отсутствует, его значение принимается по умолчанию согласно спецификации.

Вот основные параметры тега <BODY>:

- text — устанавливает цвет текста документа; значение цвета представлено в виде RRGGBB (например, text="000000" — черный цвет);
- link — устанавливает цвет гиперссылок, значение цвета имеет вид RRGGBB (например, text="FF0000" — красный цвет);

- `vlink` — устанавливает цвет гиперссылок, на которых пользователь уже побывал (например, `text="00FF00"` — зеленый цвет);
- `alink` — устанавливает цвет гиперссылок при нажатии на них (например, `text="0000FF"` — синий цвет);
- `bgcolor` — устанавливает цвет фона документа; значение цвета представлено в виде `RRGGBB` (например, `text="FFFFFF"` — белый цвет);
- `background` — устанавливает изображение для фона документа (например, `background="bg.gif"`);
- `topmargin` — задает величину верхнего поля документа (например, `topmargin="0"`);
- `leftmargin` — задает величину левого поля документа (например, `leftmargin="10"`).

> ***Примечание***
> В HTML-коде цвет определяется 6-значным кодом `RRGGBB` (красный, красный, зеленый, зеленый, синий, синий), а в JavaScript или в CSS таким же 6-значным кодом или английским названием цвета. Со значениями кодов, обозначающих цвета, можно ознакомиться по следующей ссылке: **https://www.rapidtables.com/web/color/html-color-codes.html**.

Вот пример использования в теге `<BODY>` указанных параметров:

```
<BODY text="black" bgcolor="white">
```

Несмотря на то, что описанные здесь параметры форматирования еще весьма широко распространены, нужно воздерживаться от их применения, так как для этих целей предназначены средства каскадных таблиц стилей, о чем речь пойдет в последующих разделах.

2.2.2. Списки

Современным стандартом HTML предусмотрены три основных вида списков:

- маркированные списки (unordered list);
- нумерованные списки (ordered list);
- списки определений (definition list).

Маркированные списки названы так потому, что перед каждым пунктом таких списков устанавливается тот или иной *маркер*. Иногда, прибегая к дословному переводу, их также называют *неупорядоченными списками* (unordered list). Маркированные списки задаются при помощи тегов `<ul>...<ul>`. Для задания элементов списка (item list) используются теги `<il>...</il>`. Например:

```
<ul>
<li>Пункт 1</li>
<li>Пункт 2</li>
<li>Пункт 3</li>
</ul>
```

Помимо элементов списка внутри тегов `<ul>...<ul>` можно размещать и другие теги — например, теги заголовков:

```
<ul>
<h3>Маркированный список</h3>
<li>Пункт 1</li>
<li>Пункт 2</li>
<li>Пункт 3</li>
</ul>
```

Тип маркера можно задавать с помощью атрибута `type`. Три его возможных значения:

- `circle` (не закрашенный кружок);
- `disk` (закрашенный кружок);
- `square` (закрашенный квадрат).

По умолчанию задан тип `disk`. Следует отметить, что различные модели браузеров могут по-разному отображать маркеры. Вот пример использования маркера типа `circle`:

```
<ul type="circle">
<li>Пункт 1</li>
<li>Пункт 2</li>
<li>Пункт 3</li>
</ul>
```

В *нумерованных списках* (ordered list), которые иногда называют *упорядоченными*, каждому пункту присваивается номер. Создаются такие списки при помощи тегов `<ol>...</ol>`. Для элементов нумерованных списков, как и в случае маркированных списков, также используются теги `<il>...</il>`. В таких списках, как и в маркированных, доступны пять типов маркеров, определяемых при помощи атрибута `type`, который может принимать следующие значения:

- `1` — арабские цифры;
- `i` — строчные римские цифры;
- `I` — прописные римские цифры;
- `a` — строчные латинские буквы;
- `A` — прописные латинские буквы.

Далее приведены примеры нумерованных списков:

```
<ol>
<h3>Нумерованный список</h3>
<li>Пункт 1</li>
<li>Пункт 2</li>
<li>Пункт 3</li>
</ol>
<ol type="I">
<li>Пункт 1</li>
<li>Пункт 2</li>
<li>Пункт 3</li>
</ol>
```

Списки определений (definition list) предназначены для того, чтобы организовать текст по примеру словарных статей. Они задаются с помощью тегов `<dl>...</dl>`, а определяемый термин или понятие (definition term) помещается в теги `<dt>...</dt>`. Определение понятия (definition description) заключается в теги `<dd>...</dd>`. В тексте, содержащемся внутри тегов `<dt>...</dt>`, недопустимы теги уровня блока, такие как `<p>` или `<div>`. Как и в предыдущих случаях, внутри списков определений могут быть теги заголовков и прочие теги:

```
<dl>
<h3>Список определений</h3>
<dt>Понятие 1</dt>
<dd>Определение понятия 1</dd>
<dt>Понятие 2</dt>
<dd>Определение понятия 2</dd>
</dl>
```

Как маркированные, так и нумерованные списки можно вкладывать друг в друга, причем допускается произвольное вложение различных типов списков. При вложении друг в друга маркированных и нумерованных списков следует быть внимательным. В листинге 2.4 приведен пример вложенных списков.

Листинг 2.4. Пример вложенных списков

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.4.

2.2.3. Таблицы

Таблицы — один из основных элементов, предназначенных для структурирования информации в HTML-документах. Хотя сейчас такое использование таблиц признано устаревшим и не рекомендуемым, его до сих пор применяют многие веб-дизайнеры. Таблица создается при помощи тега `<TABLE>`, который может иметь следующие параметры:

- `align` — задает выравнивание таблицы (`align=left/center/right`);
- `border` — задает толщину линий таблицы (в пикселах);
- `bgcolor` — устанавливает цвет фона документа;
- `background` — устанавливает изображение фона документа;

- cellpadding — задает ширину промежутков между содержимым ячейки и ее границами (в пикселах), т. е. определяет поля внутри ячейки;
- cellspacing — задает ширину промежутков между ячейками (в пикселах);
- width — задает ширину таблицы в пикселах, процентах или частях.

Для создания заголовка таблицы служит тег <CAPTION>. По умолчанию заголовки центрируются и размещаются либо над (<CAPTION align="top">), либо под таблицей (<CAPTION align="bottom">). Заголовок может состоять из любого текста и изображений. Текст будет разбит на строки, соответствующие ширине таблицы. Каждая новая строка таблицы создается тегом <TR> (Table Row), который может иметь дополнительные параметры:

- align — задает горизонтальное выравнивание информации в ячейках (align= left/cente/|right/justify);
- valign — задает вертикальное выравнивание информации в ячейках (valign=top/ middle/bottom).

Внутри строки таблицы размещаются ячейки с данными, создаваемые тегами <TD>. Число тегов <TD> в строке определяет число ячеек (столбцов) таблицы. Тег <TD> может иметь следующие дополнительные параметры:

- align — задает горизонтальное выравнивание информации в ячейке (align=left/ center/right/justify);
- valign — задает вертикальное выравнивание информации в ячейке (valign=top/ middle/bottom);
- colspan — объединение столбцов в строке (например, colspan=2);
- rowspan — объединение строк в столбце (например, rowspan=3);
- width — задает ширину ячейки в пикселах или процентах;
- nowarp — запрещает переход текста в ячейке на новую строку.

Для задания заголовков столбцов и строк таблицы служит тег заголовка <TH> (Table Header). Этот тег аналогичен тегу <TD> с той лишь разницей, что текст в теге <TH> по умолчанию выделяется полужирным шрифтом и располагается по центру ячейки.

В листинге 2.5 приведен пример простой таблицы.

Листинг 2.5. Пример таблицы

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.5.

Благодаря наличию большого числа параметров, а также возможности создания границ нулевой толщины, таблица может выступать в роли невидимой модульной сетки, относительно которой добавляется текст, изображения и другие элементы, организуя информацию на странице.

Можно отметить следующие возможности при табличной верстке страниц:

- создание колонок;
- создание "резинового" макета;
- "склейка" изображений;
- включение фоновых рисунков;
- выравнивание элементов.

При этом табличная верстка имеет и ряд недостатков:

- долгая загрузка;
- громоздкий код;
- плохая индексация поисковиками;
- нет разделения содержимого и оформления;
- несоответствие стандартам.

Рассмотрим некоторые типовые модульные сетки.

Двухколонная модульная сетка часто применяется на небольших информационных сайтах. Как правило, в первой колонке располагается логотип и меню сайта, а во второй — основной материал. Пример такой модульной сетки приведен на рис. 2.5.

Рис. 2.5. Двухколонная модульная сетка

В листинге 2.6 приведен HTML-код, реализующий эту структуру.

Листинг 2.6. Пример HTML-кода двухколонной модульной сетки

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.6.

Трехколонная модульная сетка применяется на крупных порталах с множеством информационных сервисов. Как правило, в первой колонке располагается меню сайта и дополнительная информация, во второй — основной материал, в третьей — дополнительные функции. Часто в модульную сетку сайта добавляют блоки "Заголовок сайта" и "Окончание сайта" (этот блок еще называют "подвалом"). Пример такой модульной сетки приведен на рис. 2.6.

В листинге 2.7 приведен HTML код, реализующий эту структуру.

Листинг 2.7. Пример HTML-кода трехколонной модульной сетки

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.7.

Рис. 2.6. Трехколонная модульная сетка

2.2.4. Тег *div*

Буквально 2–3 года назад "скелетом" для сайта служила таблица (тег `<table>`). Создавалась одна большая таблица, которая потом делилась на несколько областей: заголовок, левый блок, правый блок, центр и низ. С появлением тега `<div>` все веб-мастера стали пользоваться им для создания "скелета" будущего сайта.

Тег разделения HTML-страниц, сокращенно называемый "div", представляет собой специальный элемент, который позволяет объединять похожие наборы содержимого веб-странице в условные группа. Элемент `<div>` появился в стандарте HTML 3.0 и представлял собой "участок" контента. Его можно

использовать как универсальный контейнер для связывания похожего содержимого. В первую очередь тег div применятся для группировки похожего содержимого, чтобы его можно было легко стилизовать. Отличный пример —группировка различных разделов веб-страницы с помощью . Например, можно объединить верхнюю навигационную панель и нижний колонтитул страницы в отдельный тег div, чтобы эти разделы можно было стилизовать вместе. Вот синтаксис тега div:

```
<div class="Имя класса">
...
</div>
```

Этот тег имеет ряд атрибутов:

- align="параметр" — задает выравнивание;
- class="имя" — определяет принадлежность к классу;
- style="стили через запятую" — задает стили;
- title="текст" — задает всплывающая подсказка к тегу.

Атрибут align (выравнивание) может принимать следующие значения:

- center — выравнивание текста по центру;
- left — выравнивание текста по левому краю;
- right — выравнивание текста по правому краю;
- justify — выравнивание по ширине.

В листинге 2.8 приведен пример использования тега .

**Листинг 2.8. Пример использования тега **

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.8.

В данном примере текст, заключенный в тег , будет выровнен по центру страницы. Более детально использование этого тега будет показано на примерах в разделе, посвященном фреймворку Bootstrap.

2.2.5. Гиперссылки

Для связи различных документов HTML предусматривает использование *ссылок*. Сам термин HTML (Hyper Text Markup Language) подразумевает их широкое применение. Для реализации ссылок в HTML служит тег <a>..., который, как и большинство HTML-тегов, является контейнерным. Основной атрибут этого тега — href, собственно и содержащий адрес ресурса, на который указывает ссылка. Внутри тега <a>... помещается текст ссылки.

В ссылках возможна как относительная, так и абсолютная адресация. При абсолютной адресации атрибуту href присваивается абсолютный URL-адрес ресурса:

```
<a href="http://server.com/doc3.htm">Ссылка на документ с абсолютным адресом
http://server.com/doc3.htm</a>.
```

При относительной адресации указывается путь к документу относительно текущей страницы:

```
<a href="doc1.htm">Ссылка на документ с относительным адресом doc1.htm</a>.
```

Если в заголовочной части документа указан тег <base>, то отсчет будет вестись от адреса, заключенного в этом теге.

Помимо веб-страниц, допускается ссылаться и на другие интернет-ресурсы: e-mail, ftp, Ghofe, WAIS, Telnet, newsgroup. Далее приведен пример ссылки на адрес электронной почты:

```
<a href="mailto:sss@mail.ru">Ссылка на адрес электронной почты sss@mail.ru</a>.
```

В роли ссылок могут выступать и рисунки. Для этого сначала нужно вставить рисунок с помощью тега <img>. У атрибута src этого тега устанавливается значение, соответствующее имени файла рисунка:

```
<img src="fig1.jpg">.
```

Далее, рисунок "обертывается" в тег ссылки:

```
<a href="link2.htm"><img src="fig1.jpg" border="0"></a>.
```

2.3. Каскадные таблицы стилей (CSS)

Если HTML служит для логического форматирования документа, то для управления его отображением на экране монитора или выводом на принтер применяются каскадные таблицы стилей (CSS). Технология CSS реализует концепцию "Документ — представление" и позволяет отделять оформление HTML-документа от его структуры. Кроме того, CSS существенно расширяет возможности представления (оформления) документов, внося множество новых возможностей.

Для того чтобы таблица стилей влияла на вид HTML-документа, она должна быть подключена к нему. Подключить каскадные таблицы стилей можно с использованием внешних, внутренних или локальных таблиц стилей:

- внешние таблицы стилей, оформленные в виде отдельных файлов, подключаются к HTML-документу при помощи тега в заголовке документа. Например:

```
<LINK rel="stylesheet" href="style.css" type="text/css">   .
```

- внутренние таблицы стилей в составе HTML-документа помещаются в заголовок страницы при помощи тега `<STYLE>`. Например:

```
<HEAD>
 <STYLE>
    P {color: #FF0000}
 </STYLE>
</HEAD>
```

- локальные таблицы стилей объявляются непосредственно внутри тега, к которому они относятся, при помощи параметра `style`. Например:

```
<P style="color: #FF0000">Каскадные таблицы стилей</p>.
```

Таблицы стилей записываются в виде последовательности тегов, для каждого из которых в фигурных скобках указывается его свойства по схеме `параметр: свойство`. Параметры внутри фигурных скобок разделяются точкой с запятой. Например:

```
P {color: #CCCCCC; font-size: 10px}
H1{color: #FF0000}.
```

С помощью каскадных таблиц стилей можно менять свойства следующих элементов: фона и цвета, шрифта, текста, полей и отступов, границ.

Для задания свойств фона и цвета служат следующие параметры:

- `background-color` — цвет заднего плана;
- `background-image` — изображение заднего плана;
- `background-repeat` — дублирование изображения заднего плана (значения: `repeat/repeat-x/repeat-y/no-repeat`);
- `background-attachment` — фиксация изображения заднего плана (значения: `scroll/fixed`);
- `background-position` — начальное положение изображения заднего плана.

Для задания свойств шрифта служат следующие параметры:

- `font-style` — стиль шрифта (значения: `normal / italic`);
- `font-weight` — начертание шрифта (значения: `normal / bold`);
- `font-size` — размер шрифта;
- `font-family` — список имен шрифтов в порядке их приоритета.

Для задания свойств текста служат следующие параметры:

- `word-spacing` — установка промежутка между словами;
- `letter-spacing` — установка высоты строки;
- `text-indent` — установка абзацного отступа;
- `text-align` — выравнивание текста;
- `vertical-align` — установка вертикального выравнивания текста;
- `text-decoration` — преобразование текста (значение: `none/underline/overline/line-through/blink`).

Для задания свойств полей и отступов служат следующие параметры (`margin` — отступы между элементами, `padding` — отступ от содержимого элемента до его границы):

- `margin-top` — установка верхнего поля;
- `margin-right` — установка правого поля;
- `margin-bottom` — установка нижнего поля;
- `margin-left` — установка левого поля;
- `margin` — установка всех полей (например: `margin: 10px 10px 10px 0px`);
- `padding-top` — установка верхнего отступа;
- `padding-right` — установка правого отступа;
- `padding-bottom` — установка нижнего отступа;
- `padding-left` — установка левого отступа;
- `padding` — установка всех отступов (например: `padding: 10px 10px 10px 0px`).

Для задания свойств границ служат следующие параметры:

- `border-width` — установка ширины границы;
- `border-color` — установка цвета границы;
- `border-style` — установка стиля границы (значения: `none/dotted/dashed/solid/double`).

Для придания управлению элементами HTML большей гибкости используются *классы*, которые позволяют задавать различные стили для одного и того же тега.
В таблицах стилей имя класса записывается после тега и отделяется от него точкой. Например:

```
P.green {color: #00FF00}
P.blue {color: #0000FF}
```

В HTML-коде соответствие тега определенному классу указывается при помощи параметра `class`. Например:

```
<p class="green">Зеленый текст</p>
<p class="blue">Синий текст</p>
```

2.4. Возможности использования JavaScript

Чаще всего код на языке JavaScript встраивается в веб-страницы для получения программного доступа к их элементам. Сценарии JavaScript являются основой клиентской части веб-приложений. Они загружаются с сервера вместе с веб-страницами и выполняются браузером на компьютере пользователя. Обработка сценариев JavaScript осуществляется встроенным в браузер интерпретатором. Что может делать JavaScript?

- В первую очередь JavaScript умеет отслеживать действия пользователя (например, реагировать на щелчок мыши или нажатие клавиши на клавиатуре, на перемещение курсора, на скроллинг).
- Менять стили, добавлять (удалять) HTML-теги, скрывать (показывать) элементы.
- Посылать запросы на сервер, а также загружать данные без перезагрузки страницы (эта технология называется AJAX).

JavaScript — это объектно-ориентированный язык. Он поддерживает несколько встроенных объектов, а также позволяет создавать или удалять свои собственные (пользовательские) объекты. Объекты JavaScript могут наследовать свойства непосредственно друг от друга, образуя цепочку "объект -> прототип".

Сценарии JavaScript бывают встроенные, т. е. их содержимое является частью документа, и внешние, хранящиеся в отдельном файле с расширением js. Сценарии можно внедрить в HTML-документ следующими способами:

- в виде гиперссылки;
- в виде обработчика события;
- внутрь элемента `<script>`.

Рассмотрим эти способы подробнее.

Если подключать JavaScript в виде гиперссылки, то для этого код скрипта нужно разместить в отдельном файле, а ссылку на файл включить либо в заголовок:

```
<head>
   <script src="script.js"></script>
</head>
```

либо в тело страницы:

```
<body>
   <script src="script.js"></script>
</body>
```

Этот способ обычно применяется для сценариев большого размера или сценариев, многократно используемых на разных веб-страницах.

Можно подключать JavaScript к обработчику события. Дело в том, что каждый HTML-элемент может иметь JavaScript-события, которые срабатывают в определенный момент. Нужно добавить необходимое событие в HTML-элемент как атрибут, а в качестве значения этого атрибута указать требуемую функцию. Функция, вызываемая в ответ на срабатывание события, и станет обработчиком события. В результате срабатывания события исполнится связанный с ним код. Этот способ применяется в основном для коротких сценариев — например, можно установить смену цвета фона при нажатии на кнопку (листинг 2.9).

Листинг 2.9. Пример обработки события

В демо-версии книги листинги программ удалены

Примечание.
В интерактивной цифровой книге можно выполнить код листинга 2.9.

Код внутри элемента `<script>` можно вставлять в любое место документа. Код, который выполняется сразу после прочтения браузером или содержит описание функции, которая выполняется в момент ее вызова, располагается внутри тега. Описание функции можно располагать в любом месте — главное, чтобы к моменту ее вызова код функции уже был загружен.

Обычно код JavaScript размещается в заголовке документа (в элементе `<head>`) или после открывающего тега `<body>`. Если скрипт используется после загрузки страницы (например, код счетчика), то его лучше разместить в конце документа:

```
<footer>
   <script>
      document.write("Введите свое имя");
   </script>
</footer>
</body>
```

В примерах этой книги мы не будем писать собственные скрипты JavaScript, поскольку подключим готовые скрипты фреймворка Bootstrap, однако полезно знать о наличии этого языка и его возможностях при создании веб-страниц.

2.5. Краткие итоги

В этой главе мы познакомились с базовыми сведениями о веб-программировании и основными языками, с помощью которых можно разрабатывать клиентскую часть веб-приложения. Процедуры создания клиентской части веб-приложений значительно упрощаются, если использовать специализированные фреймворки для frontend-программирования. Наиболее популярным среди программистов является фреймворк Bootstrap, с которым мы более детально познакомимся в следующей главе.

Приложение. Порядок работы с интерактивной цифровой книгой

1. Порядок работы с демо-версией интерактивной электронной книги по фреймворку Django

Интерактивная электронная книга представляет собой web приложение, благодаря которому можно изучать текст книги, просматривать и копировать листинги программ, запускать программные модули на выполнение. Если пользователь не был зарегистрирован на сайте и не приобретал книгу, то для него на главной странице сайта «Академия Python» будет доступна только демо-версия интерактивной электронной книги (рис. П.1).

Рис. П.1. Карточка с демо-версией интерактивной электронной книги по фреймфорку Django на главной страница сайта «Академия Python»

В демо-версии книги нет возможности запускать примеры программных модулей на выполнение, для пользователя будут доступны только первые две главы в виде PDF-файла. Чтобы открыть вкладку с демо-версией книги нужно на карточке с обложкой книги нажать кнопку «**Демо версия (1-я и 2-я глава книги**» (рис. П.2).

Рис. П.2. Кнопка для открытиы вкладки с демо-версией интерактивной электронной книги

После этого в браузере пользователя откроется новая вкладка, в которой будет представлен текст книги в формате PDF (рис. П.3).

Рис. П.3. Вкладка интернет браузера с текстом интерактивной эдектронной книги

Данная вкладка разбита на три части:

- В верхней части окна находится строка с элементами главного меню;
- В левой части окна находится боковая панель с элементами бокового меню;
- В центре окна находятся страницы с текстом книги.

В строке главного меню имеется несколько кнопок:

- В верхнем левом углу кнопка с тремя полосками (скрыть/показать боковое меню);
- В центре окна кнопки «+», «-» для увеличения, уменьшения текста ниги;
- В правом верхнем углу кнопка со стрелкой для скачивания PDF-файла;
- В правом верхнем углу кнопка с изображением принтера для вывода текста книги на печать.

В левой боковой панели имеются две кнопки. Верхняя кнопка служит для включения режима показа макета страниц и быстрого их перелистывания (рис. П.4).

Рис. П.4. Кнопка для включения режима быстрого перелистывания страниц текста

В этом режиме в боковой панели показаны уменьшенные макеты страниц текста. Текущая страница, которая имеет более яркий фон, будет отображаться в центральном окне. Пользователь имеет возможность с помощью мыши перемещать бегунок, расположенный на данной панели, тем самым перелистывать страницы текста. После выделения мышью одной из страниц этой панели она будет показана в центральной области экрана (рис. П.5).

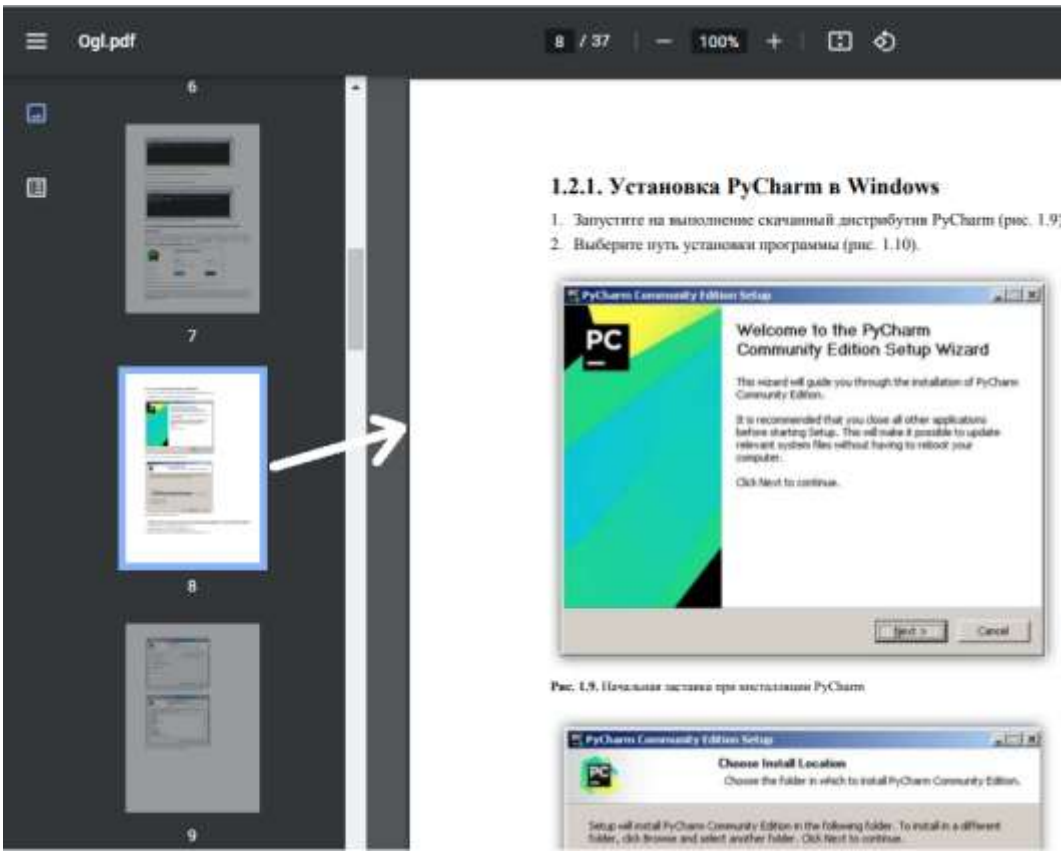

Рис. П.5. Вкладка браузера в режиме быстрого перелистывания страниц

Нижняя кнопка в левой боковой панели служит для включения режима показа оглавления книги. В этом режиме в левой боковой панели появляется оглавление книги (рис. П.6).

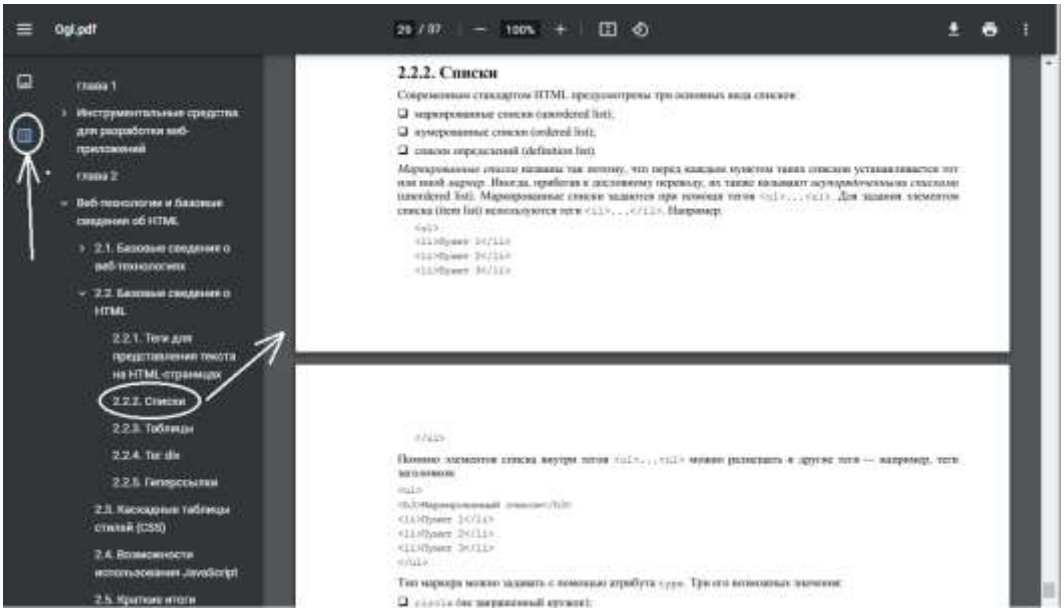

Рис. П.6. Вкладка браузера в режиме показа оглавления книги

В этом режиме пользователь может в оглавлении с помощью мыши открывать (закрывать) любую главу и любой раздел книги, при этом в центральной части окна будет появляться первая страница соответствующего раздела.

Кроме того, пользователь имеет возможность в окне браузера открыть несколько вкладок с текстом книги, в каждой вкладке открыть разные разделы и, переключаясь между вкладками, просматривать разные разделы книги без перелистывания большого количества страниц.

2. Порядок работа с полной версией интерактивной электронной книги по фреймворку Django

Для того, чтобы получить полный доступ к интерактивной цифровой книге нужно зарегистрироваться на сайте и приобрести данную книгу. Для регистрации на сайте нужно на левой боковой панели нажать на ссылку «**Вход**». После этого откроется окно – приглашение, на котором нужно нажать на ссылку «**Регистрация**» (рис. П.7).

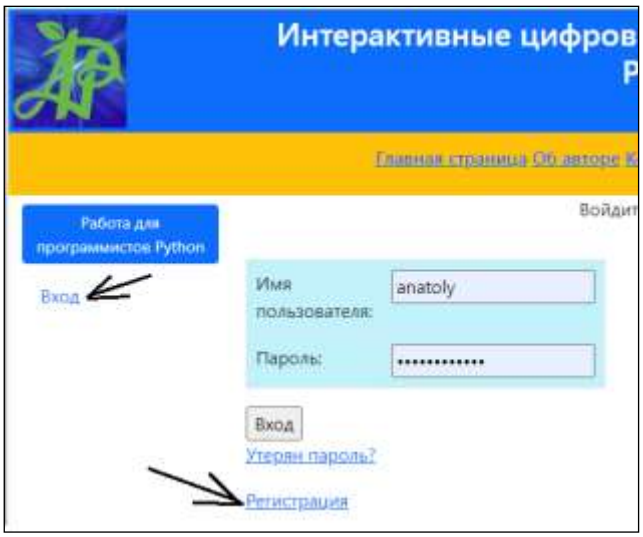

Рис. П.7. Ссылка для открытия окна регистрации пользователя на сайте

После этого откроется окно, в котором пользователю необходимо ввести свои регистрационные данные (рис. П.8).

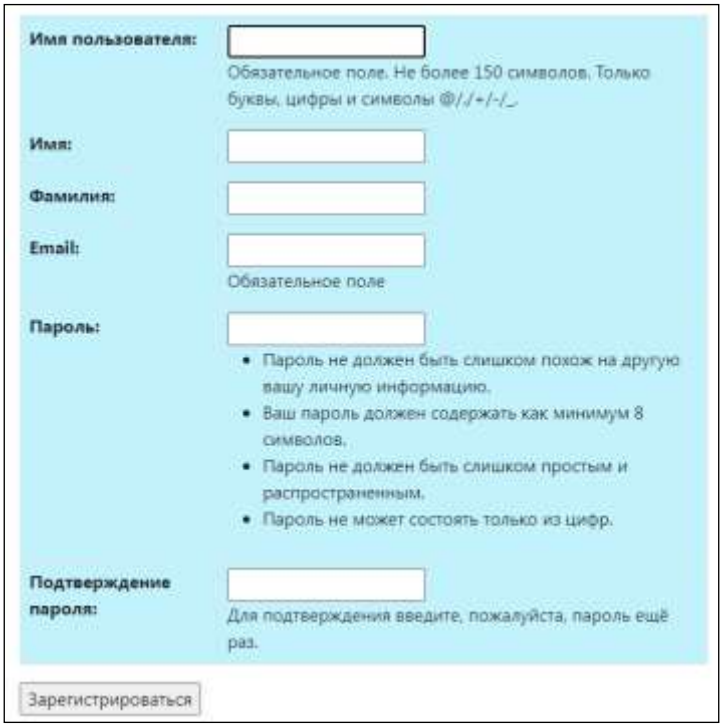

Рис. П.8. Окно для ввода регистрационных данных

После того, как пользователь зарегистрируется на сайте и оплатит книгу, он получит полный доступ ко всем разделам книги, к программному коду и модулям, которые позволяют запускать примеры программ. Для того, чтобы открыть интерактивную электронную книгу для чтения нужно в боковой панели сайта выбрать опцию «Мои покупки», после этого откроется страница со сведениями о приобретенных товарах (рис. П.9).

Рис. П.9. Страница со сведениями о приобретенных товарах

На данной странице нужно открыть выпадающий список «Категории покупок» и выбрать опцию «Книги» (рис. П.10).

Рис. П.10. Выбор опции «Книги» на странице со сведениями о приобретенных товарах

После этого откроется страница со списком приобретенных книг. Если пользователь приобрел книгу по фреймворку Django, то на этой странице будет представлена карточка с данной книгой (рис. П.11).

Рис. П.11. Карточка с интерактивной электронной книгой по фреймворку Djangox

На карточке с интерактивной электронной книгой нужно нажать на кнопку «Открыть книгу на сайте». После этого откроется новая страница, которая будет являться главной страницей полной web версией книги по фреймворку Django (рис. П.12).

Рис. П.12. Главная страница полной web версиии книги по фреймворку Django

На данной странице в строке главного меню имеется всего две опции:

- Главная страница – возврат к главной странице сайта «Академия Python»;
- Книга Django – обновление текущей страницы с полной web версией интерактивной электронной книги по Django.

В левой боковой панели появятся следующие элементы, которые позволят начать изучение материалов книги:

- ❑ Кнопка для открытия вкладки браузера с полным текстом книги «Показать текст книги»;
- ❑ Ссылки на листинги программных модулей, которые приведены в книге.

С текстом книги пользователь может работать в том же режиме, который был описан в предыдущем разделе для демо-версии книги. При нажатии на кнопку «Показать текст книги» будет открыта новая вкладка, в которой будет показан полный текст книги в виде PDF-файла (рис. П.13).

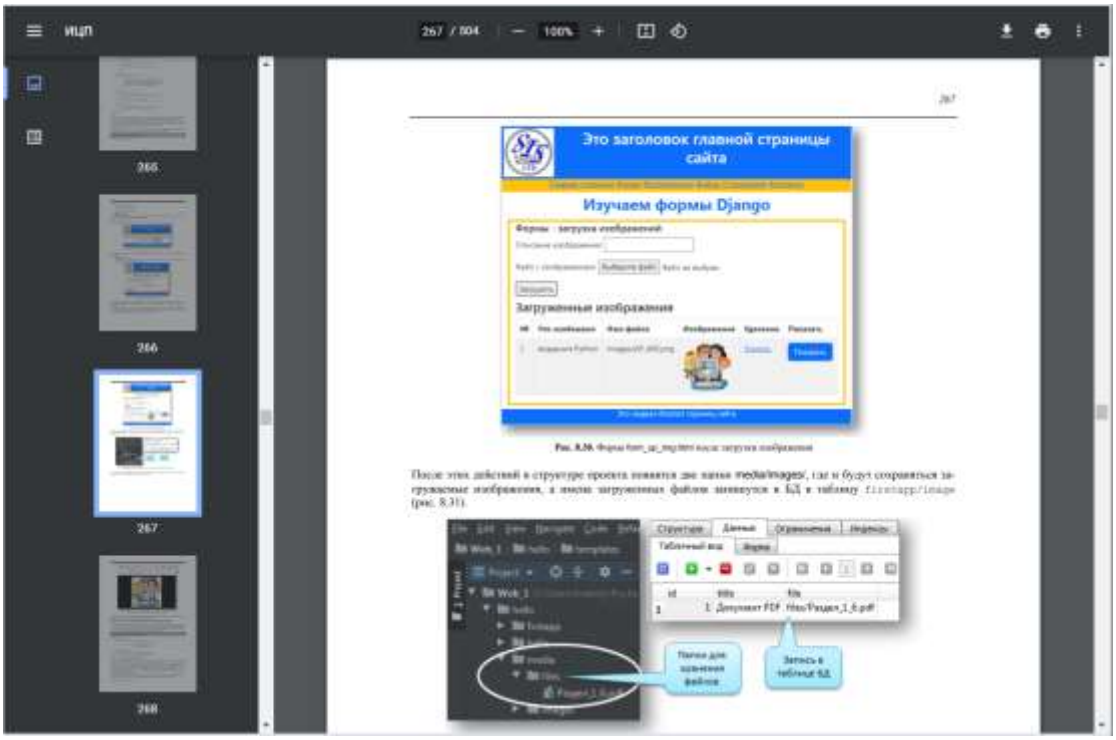

Рис. П.13. Вкладка с полным текстом книги по фреймворку Django

На данной вкладке пользователь может, либо напрямую пролистывать и просматривать нужные страницы книги, либо перемещаться между разделами через оглавление книги. Теперь пользователь может в одной вкладке работать с текстом книги, а в другой вкладке изучать листинги программ, копировать программный код и запускать примеры программных модулей на выполнение.

Для запуска того или иного программного модуля нужно в боковой панели найти интересующую главу и, кликнув на ней мышью открыть список листингов программных модулей, относящихся к данной главе (рис. П.14).

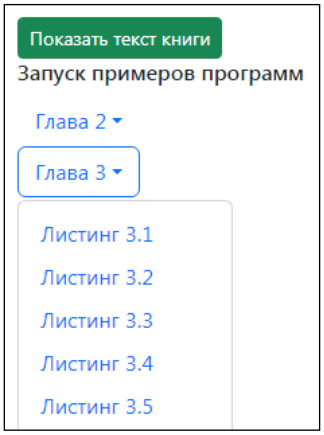

Рис. П.14. Раскрывающийся список с листингом программ

После того, как пользователь выбрал нужный листинг, откроется страница с соответствующим программным кодом и кнопкой для запуска этого программного кода на выполнение (рис. П.15).

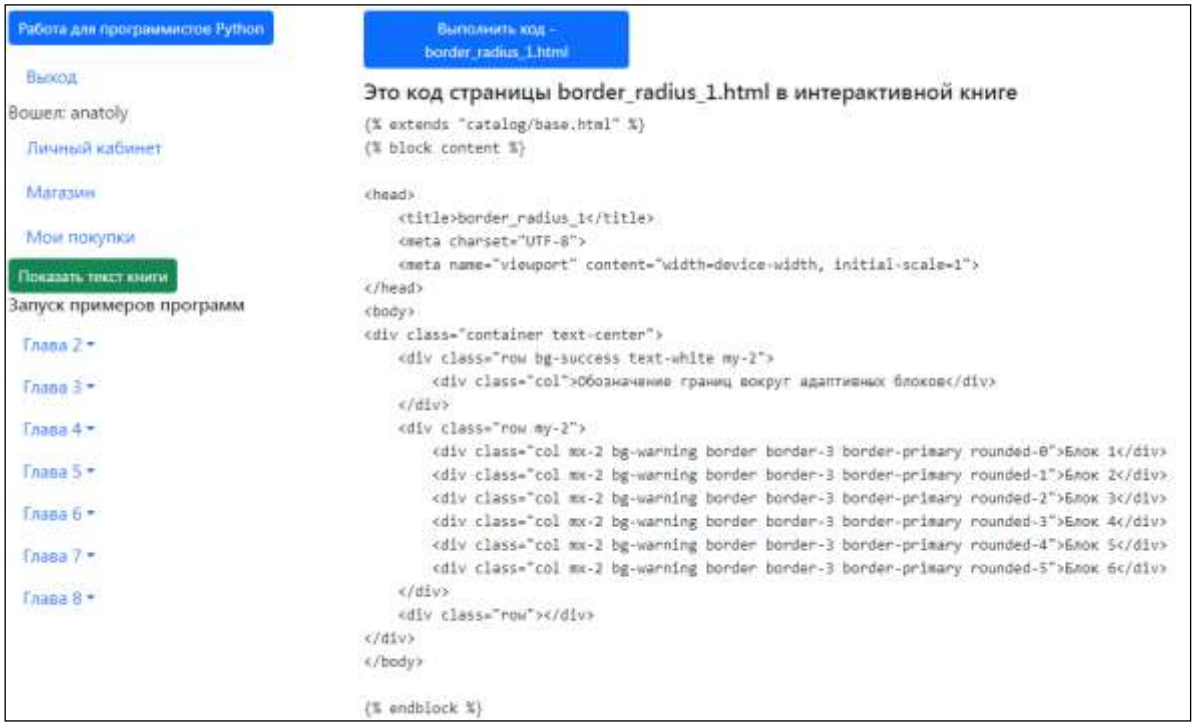

Рис. П.15. Листинг программного модуля с кнопкой запуска модуля на выполнение

На данной странице пользователь может изучить данный программный код и скопировать его на свой компьютер. Следует отметить, что этот код с детальными пояснениями и описанием его фрагментов приведен в тексте книги. После нажатия на кнопку «**Выполнить код**» в верхней части страницы будут показаны результаты работы этого программного модуля (рис. П.16).

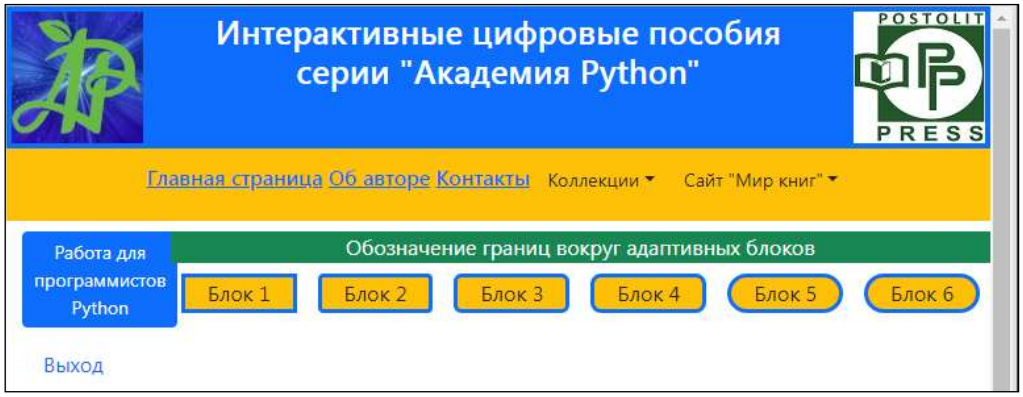

Рис. П.16. Результаты работы программного модуля на странице сайта

Таким образом, пользователь может, не устанавливая на свой компьютер никаких дополнительных программных средств, фреймворков и библиотек, посмотреть на результаты работы приведенных в книге примеров программных модулей. Следует более подробно остановиться на запуск примеров программных модулей для глав 7 и 8.

В главе 7 левой боковой панели показаны примеры обработки различных полей в формах Django. При выборе главы 7 в боковой панели откроется всего одна опция для запуска листингов 7.1-7.6 (рис.П.17).

Рис. П.17. Опция выбора листингов программ для главы 7 в боковом меню сайта

При выборе этой опции откроется страница с этими листингами и кнопкой запуска листинга 7.1 (рис.П.18).

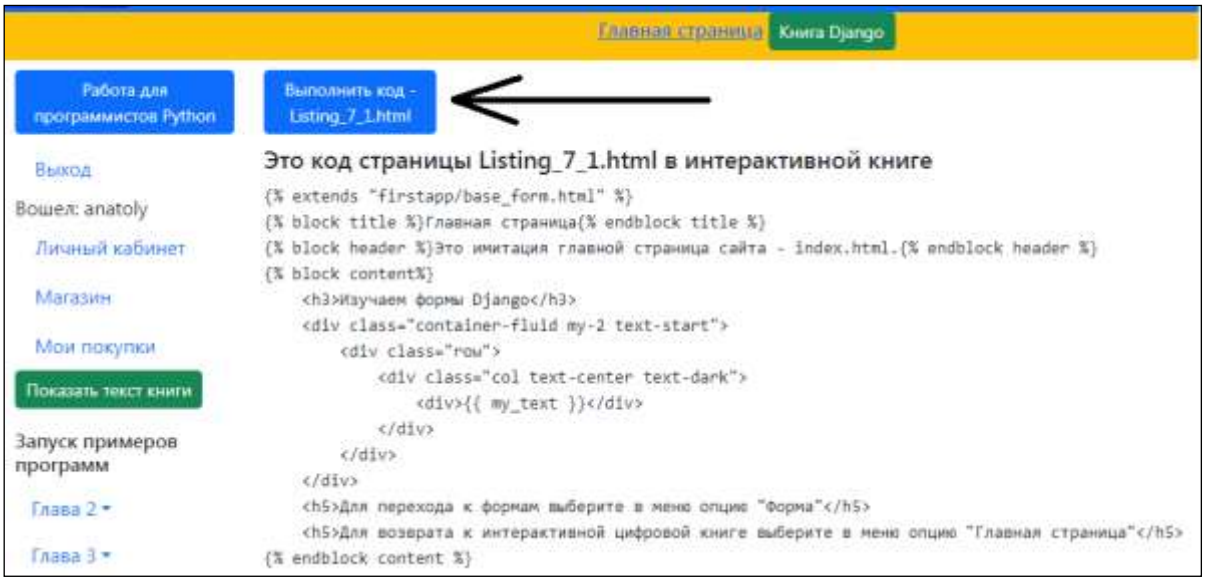

Рис. П.18. Кнопка для запуска листинга 7.1.

После нажатия на данную кнопку откроется страница с имитацией главной страницы учебного сайта, при этом в верхней строке меню появится опция «Форма» (рис.П.19).

Рис. П.19. Опция меню для запуска программных модулей с формами главы 7

После выбора опции меню «Формы» откроется страница для изучения полей форм Django с кнопкой для выбора типа поля (рис. П.20).

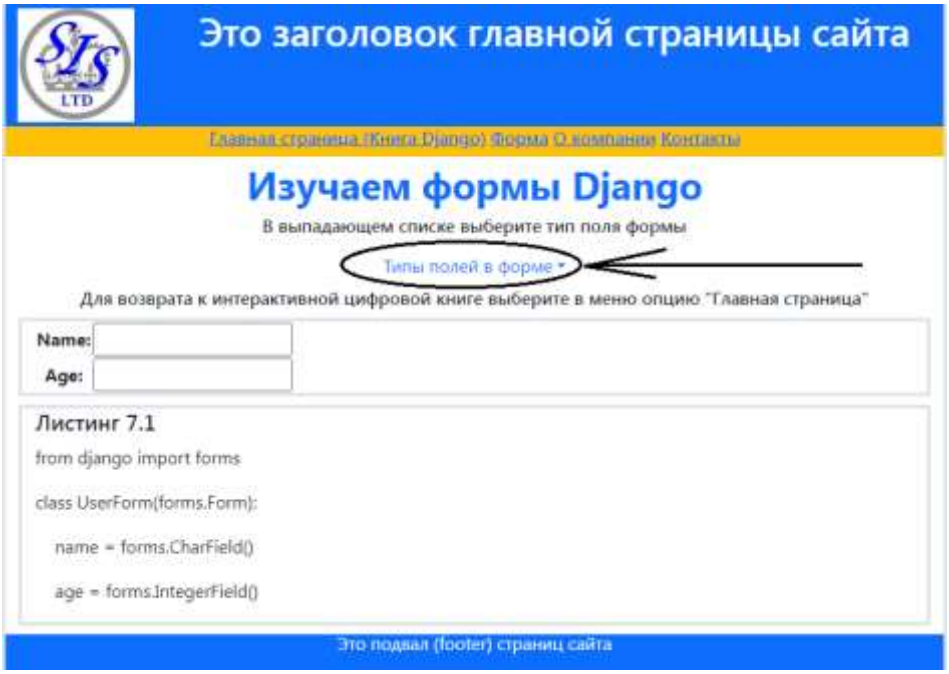

Рис. П.20. Страница с выпадающим меню для выбора типа поля формы Django

В раскрывающемся списке «Типы полей формы» можно выбрать один из листингов с тем или иным типом поля (рис. П.21).

Рис. П.21. Раскрывающийся список для выбора формы с определенным типом поля

После выбора того или иного типа поля откроется соответствующая форма Django, и пользователь может посмотреть как это поле будет выглядеть и работать в реальном приложении. Пример отображения формы с полем типа ChoiceField приведен на рис. П.22.

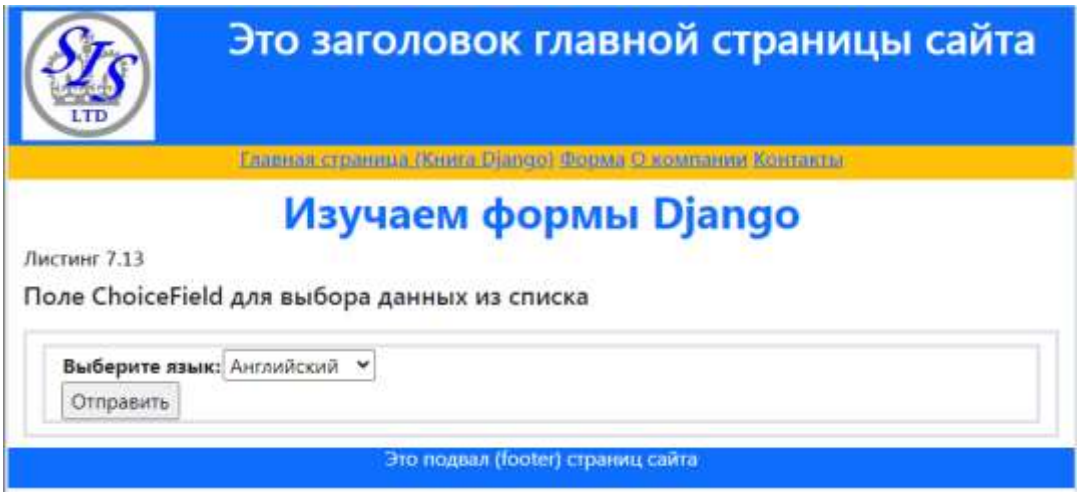

Рис. П.22. Пример страницы с полем ChoiceField формы Django

Для выбора другой формы с другим типом поля нужно в верхнем меню выбрать опцию «Форма». Для возврата к главной странице книги по фреймворку Django нужно в верхнем меню выбрать опцию «Главная страница (Книга Django).

В главе 8 левой боковой панели показаны примеры работы с моделями данных в формах Django. При выборе главы 8 в боковой панели откроется всего одна опция для запуска листинга 8.1 (рис.П.23).

Рис. П.23. Опция выбора листингов программ для главы 8 в боковом меню сайта

При выборе этой опции откроется страница с этим листингом и кнопкой запуска листинга 8.1 (рис.П.24).

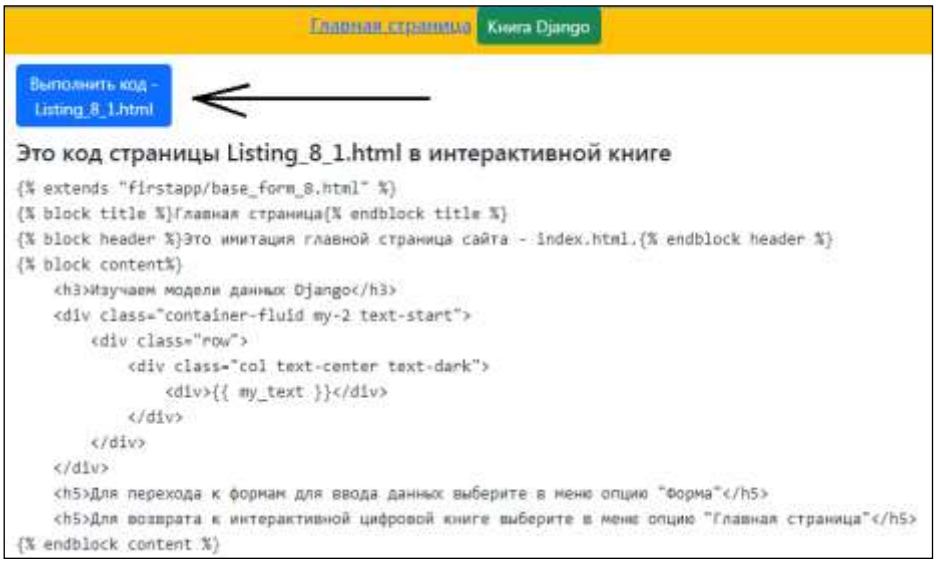

Рис. П.24. Кнопка для запуска листинга 8.1.

После нажатия на данную кнопку откроется страница с имитацией главной страницы учебного сайта, при этом в верхней строке меню появится опция «Форма» (рис.П.25).

Рис. П.25. Опция меню для запуска программных модулей с формами главы 8

После выбора опции меню «Форма» откроется страница для изучения моделей данных в формах Django с кнопкой для выбора той или иной формой Django для работы с данными через модели (рис. П.26).

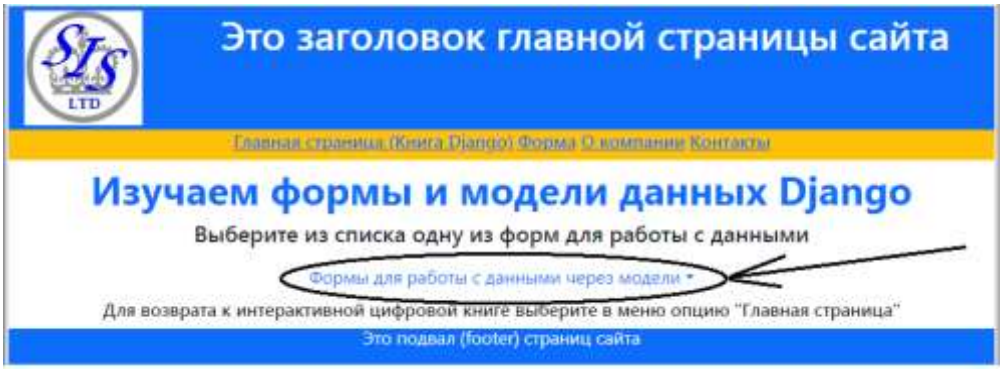

Рис. П.26. Страница с выпадающим меню для выбора типа формы Django для работы с данными через модели

В раскрывающемся списке «Формы для работы с данными через модели» можно выбрать один из листингов с тем или иным типом формы для взаимодействия с данными (рис. П.27).

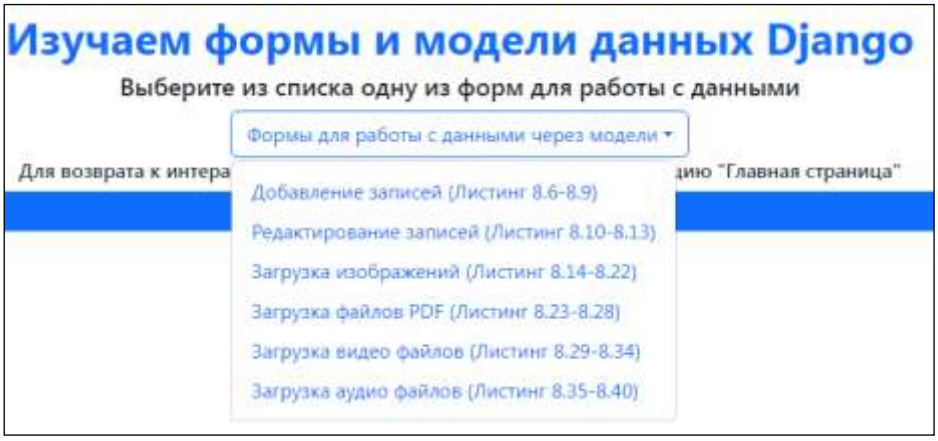

Рис. П.27. Раскрывающийся список для выбора типа взаимодействия с данными

После выбора того или иного типа формы взаимодействия с данными через модели пользователь может посмотреть как это будет выглядеть и работать в реальном приложении. Пример отображения формы для загрузки на сайт изображений приведен на рис. П.28.

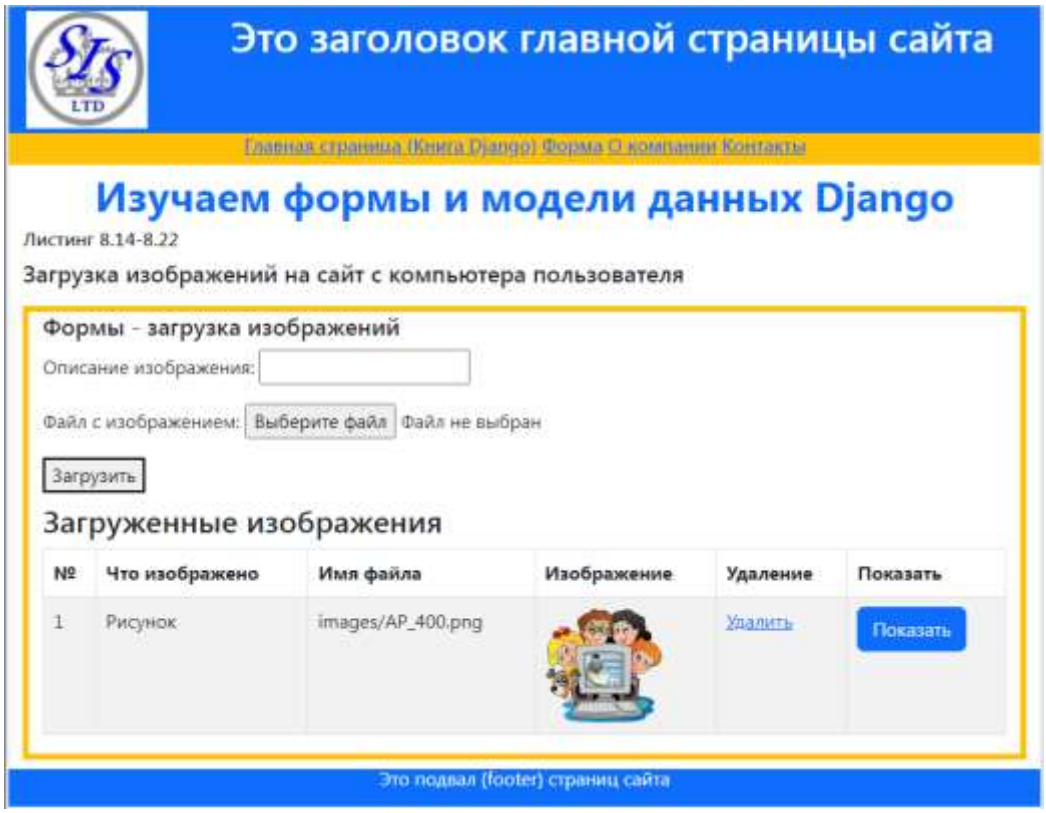

Рис. П.28. Пример страницы с формой для загрузки изображений

Для выбора другой формы работы с другими типами данных нужно в верхнем меню выбрать опцию «Форма». Для возврата к главной странице книги по фреймворку Django нужно в верхнем меню выбрать опцию «Главная страница (Книга Django).