

Оглавление

Предисловие от издательства	10
Об авторе.....	11
О рецензенте.....	12
Введение	13
Часть I. Построение приложения на Dash	17
Глава 1. Знакомство с экосистемой Dash	18
Технические требования.....	18
Настройка окружения	19
Исследование фреймворка Dash и сопутствующих пакетов	20
Пакеты, содержащиеся во фреймворке Dash	21
Введение в базовую структуру приложения Dash	22
Создание и запуск простого приложения Dash.....	23
Добавление HTML и других компонентов в приложение.....	25
Добавление компонентов HTML в приложение Dash.....	26
Проектирование макета и управление темами	28
Темы.....	29
Координатная сетка и чувствительность к изменениям.....	30
Встроенные компоненты.....	32
Кодировка цветов.....	32
Добавление компонентов Dash Bootstrap в приложение	33
Заключение	35
Глава 2. Структура приложений Dash.....	36
Технические требования.....	36
Использование Jupyter Notebook для запуска приложений Dash.....	37
Изоляция функционала для упрощения процесса разработки и отладки	37
Создание чистой функции на Python.....	39
Знакомство с параметром ID компонентов Dash	40
Использование элементов ввода и вывода.....	41
Определение ввода и вывода	42
Шаблон функции обратного вызова	43
Реализация функции обратного вызова.....	44
Внедрение функции в приложение	45
Свойства функций обратного вызова.....	53
Заключение	55
Глава 3. Работа с объектом Figure.....	56
Технические требования.....	56
Введение в объект Figure.....	57
Знакомство с атрибутом data.....	59
Знакомство с атрибутом layout.....	61
Интерактивное исследование объекта Figure	62

Опции настройки для объекта Figure	63
Способы преобразования графиков.....	64
Преобразование графиков в HTML	64
Работа с настоящим набором данных	65
Преобразование данных как важная часть процесса визуализации.....	68
Придание графику интерактивности за счет обратного вызова	69
Добавление функционала в приложение	72
Создание тем для графиков.....	74
Заключение	75
Глава 4. Подготовка и преобразование данных. Введение	
в Plotly Express	76
Технические требования.....	76
Длинный формат данных (tidy).....	77
Примеры графиков Plotly Express	77
Основные атрибуты длинного формата данных (tidy).....	80
Роль навыков в области преобразования данных.....	81
Исследование исходных файлов	82
Отмена свертывания датафреймов	91
Сведение датафреймов.....	93
Объединение датафреймов	95
Знакомство с Plotly Express.....	97
Plotly Express и объекты Figure.....	102
Создание диаграммы Plotly Express на основе набора данных	104
Добавление данных и столбцов в набор.....	106
Заклучение	107
Часть II. Расширение функционала приложений.....	109
Глава 5. Интерактивное сравнение данных при помощи	
столбчатых диаграмм и выпадающих списков	110
Технические требования.....	111
Построение вертикальных и горизонтальных столбчатых диаграмм.....	111
Создание вертикальных столбчатых диаграмм со множеством значений.....	118
Связывание столбчатых диаграмм с выпадающими списками.....	119
Разные способы отображения столбчатых диаграмм с несколькими	
рядами данных	123
Создание датафрейма с данными о доходах.....	124
Внедрение изменений в приложение.....	128
Использование ячеистой структуры для вывода множественных	
диаграмм (фасетирование).....	130
Исследование дополнительных возможностей выпадающих списков	
(множественный выбор, заместители текста и т. д.).....	133
Добавление заместителя текста для выпадающего списка	133
Изменение темы приложения.....	134
Изменение размеров компонентов	136
Заклучение	137

Глава 6. Исследование переменных при помощи точечной диаграммы и фильтрация наборов данных	139
Технические требования.....	140
Различные способы использования точечных диаграмм: маркеры, линии и текст.....	140
Маркеры, линии и текст.....	141
Отображение нескольких рядов данных на одной точечной диаграмме	144
Настройка цветов на точечной диаграмме.....	147
Дискретные и непрерывные переменные.....	147
Использование цветов с непрерывными переменными.....	148
Создание цветовых шкал вручную.....	151
Использование цветов с дискретными переменными.....	153
Управление наложениями и выбросами при помощи прозрачности, символов и масштаба.....	156
Прозрачность и размер маркеров.....	157
Использование логарифмических шкал.....	158
Знакомство со слайдерами, включая слайдеры диапазонов.....	160
Настройка подписей и значений слайдеров.....	163
Заключение.....	168
Глава 7. Работа с географическими картами и обогащение дашбордов при помощи языка разметки Markdown	169
Технические требования.....	170
Знакомство с картограммами.....	170
Использование анимации для добавления нового слоя в визуализацию....	172
Использование функций обратного вызова с картами.....	174
Создание компонента Markdown.....	177
Знакомство с проекциями карты.....	182
Использование точечных карт.....	183
Использование карт Mapbox.....	185
Другие опции и инструменты для работы с картами.....	190
Внедрение интерактивной карты в приложение.....	191
Заключение.....	192
Глава 8. Определение частотности данных с помощью гистограмм и построение интерактивных таблиц	193
Технические требования.....	194
Создание гистограммы.....	194
Настройка гистограммы, включая изменение количества столбиков и отображение множественных данных.....	195
Использование цвета для детализации гистограммы.....	197
Отображение множественных гистограмм.....	198
Добавление гистограммам интерактивности.....	201
Создание двумерной гистограммы.....	205
Создание DataTable.....	207
Настройка отображения таблицы данных (ширина и высота ячеек, отображение текста и т. д.).....	208

Добавление гистограмм и таблиц данных в приложение	210
Заключение	212
Что мы узнали из первых двух частей книги.....	214
Часть III. Развитие приложений. Новый уровень	215
Глава 9. Машинное обучение: пусть данные говорят сами за себя.....	216
Технические требования	217
Кластеризация данных.....	217
Поиск оптимального количества кластеров	221
Кластеризация стран по численности населения	224
Подготовка данных с использованием библиотеки scikit-learn.....	226
Заполнение пропущенных значений	227
Масштабирование данных при помощи scikit-learn	228
Создание интерактивного приложения с применением кластеризации по методу <i>k</i> -средних	229
Заключение	234
Глава 10. Ускорение работы приложений с помощью улучшений функций обратного вызова	235
Технические требования	236
Знакомство с элементом State.....	236
Различия между элементами Input и State	237
Создание взаимосвязанных компонентов	241
Добавление пользователем динамических компонентов в приложение	246
Введение в шаблонные обратные вызовы.....	248
Заключение	253
Глава 11. Ссылки и многостраничные приложения	255
Технические требования	256
Знакомство с компонентами Location и Link	256
Работа с компонентом Link	257
Разбор ссылок и использование их составляющих для изменения приложения	259
Адаптирование приложения под множественные макеты.....	260
Отображение содержимого на основе ссылки	263
Добавление динамически сгенерированных ссылок в приложение	264
Внедрение в приложение интерактивности на основе ссылок.....	265
Заключение	268
Глава 12. Развертывание приложения	269
Технические требования	270
Основы рабочего процесса разработки, развертывания и обновления приложений	270
Аренда виртуального сервера и настройка аккаунта	272
Подключение к серверу при помощи Secure Shell (SSH)	274
Запуск приложения на сервере.....	276

Настройка и запуск приложения через WSGI-сервер	279
Настройка и конфигурирование веб-сервера	280
Поддержка приложения и его обновление	282
Исправление ошибок и внесение изменений в приложение	282
Обновление пакетов Python	283
Поддержка сервера.....	284
Развертывание и масштабирование приложений Dash с помощью Dash Enterprise	285
Инициализация приложения	285
Создание приложения (необязательная фаза).....	286
Подготовка папки проекта	286
Развертывание приложения в Dash Enterprise.....	287
Заключение	288
Глава 13. Следующие шаги	290
Технические требования.....	290
Развитие навыков в области анализа и подготовки данных.....	291
Исследование новых техник визуализации.....	292
Знакомство с другими компонентами Dash.....	293
Создание собственных компонентов Dash.....	293
Реализация и визуализация моделей машинного обучения	294
Повышение эффективности и использование инструментов для работы с большими данными.....	294
Масштабирование с Dash Enterprise	298
Dash Design Kit.....	299
App Manager.....	299
Snapshot Engine	299
Повышение производительности с помощью Job Queue.....	300
Корпоративная безопасность.....	300
Консультационная служба	300
Заключение	300
Предметный указатель	302

Об авторе

Элиас Даббас – практикующий специалист по онлайн-маркетингу, а также обработке и анализу данных. Совместив эти области, он нашел себя в проектировании программного обеспечения с открытым кодом для создания дашбордов и приложений для работы с данными. Также он специализируется на создании программ для интернет-маркетинга.

Элиас является автором популярной библиотеки *adverttools* для Python, предлагающей богатый выбор маркетинговых инструментов с уклоном в оптимизацию поисковых систем (SEO), поисковый маркетинг (SEM), сбор данных и текстовый анализ.

О рецензенте

Леонардо Феррейра – бухгалтер, самостоятельно освоивший обработку и анализ данных до уровня Kaggle Grandmaster и выступающий разработчиком платформ в области аналитики данных. Он начал свое обучение в 2017 году и уже через несколько месяцев приступил к работе по изучаемой специальности. С тех пор Леонардо успел поработать в крупных бразильских и международных компаниях, реализовав более сотни проектов с открытым исходным кодом с портфолио на GitHub и Kaggle. Обладает статусом *Top Rated Plus* на фрилансерской платформе *Upwork*, в рамках которой реализовал более 20 проектов по анализу данных. Также интересуется решениями на базе блокчейн-платформы Cardano.

Введение

Фреймворк Dash от Plotly позволяет разработчикам Python создавать полноценные приложения для аналитической работы с данными и интерактивные дашборды. Книга, которую вы держите в руках, призвана помочь вам исследовать богатый функционал фреймворка Dash по визуализации данных и научиться извлекать максимум возможного из исходной информации.

Начнем мы с описания экосистемы Dash, основных пакетов, входящих в состав этого фреймворка, а также сторонних библиотек, позволяющих структурировать данные для вашего приложения.

После этого приступим к созданию первого приложения с использованием фреймворка Dash и добавлению в него базового функционала. Далее вы познакомитесь с такими специфическими элементами приложений, как выпадающий список, флажок, ползунок, календарь и др., а также научитесь связывать их с диаграммами и прочими элементами вывода. В зависимости от данных, которые вы визуализируете, вы будете использовать наиболее подходящие типы диаграмм, включая точечные диаграммы, линейные графики, столбчатые диаграммы, гистограммы, карты и пр., и узнаете, как можно адаптировать их под собственные нужды.

Прочитав эту книгу, вы сможете разрабатывать и развертывать сложные интерактивные дашборды, производить многоступенчатый рефакторинг кода и оптимизировать написанные вами приложения.

Для кого эта книга

Книга, которую вы начинаете читать, предназначена для специалистов по работе с данными и аналитиков, желающих больше узнать о своих исходных данных при помощи интерактивных дашбордов, включающих полный спектр визуализаций. Предполагается, что вы хотя бы на базовом уровне знаете язык программирования Python. Это поможет вам быстрее и лучше усвоить техники, описанные в книге.

Структура книги

Глава 1. Знакомство с экосистемой Dash. В данной главе вы познакомитесь с общей экосистемой фреймворка Dash, пакетами, входящими в его состав, а также сторонними библиотеками. Прочитав эту вводную главу, вы научитесь отличать разные элементы приложения и узнаете, для чего предназначен каждый из них. В качестве бонуса вы даже создадите свое первое простое приложение.

Глава 2. Структура приложений Dash. Из этой главы вы узнаете, как можно добавить созданному ранее приложению интерактивности. Здесь вы познакомитесь с концепцией *обратных вызовов* (callback) и научитесь объединять разные визуальные элементы приложения. Вы также увидите, как

с помощью функций обратного вызова можно позволить пользователю управлять содержимым одного визуального элемента посредством другого.

Глава 3. Работа с объектом Figure. В третьей главе книги вы познакомитесь с ключевым объектом *Figure*, узнаете о его компонентах, а также о способах управления им и преобразовании его в различные форматы. Позже мы используем полученные навыки для создания особых типов диаграмм для нашего приложения.

Глава 4. Подготовка и преобразование данных. Введение в Plotly Express. Здесь вы узнаете о форматах данных, наиболее пригодных для анализа. Также вы познакомитесь с пакетом Plotly Express и увидите, с какой легкостью можно с его помощью создавать диаграммы и связывать данные с элементами визуализации.

Глава 5. Интерактивное сравнение данных при помощи столбчатых диаграмм и выпадающих списков. В этой главе мы немного глубже погрузимся в функционал диаграмм и изучим дополнительные возможности визуализации данных. После этого вы узнаете, как можно дать пользователю возможность выбрать сравниваемые элементы при помощи выпадающих списков.

Глава 6. Исследование переменных при помощи точечной диаграммы и фильтрация наборов данных. В данной главе мы подробно рассмотрим один из наиболее популярных видов визуализации, а именно диаграмму рассеяния или точечную диаграмму. Как и в случае со столбчатой диаграммой, мы рассмотрим различные варианты настройки этого типа визуализации. Точечные диаграммы отличаются очень богатыми возможностями для настройки, включая управление размером точек в зависимости от выбранной переменной, исключение наложения точек друг на друга и вывод диаграммы с большим количеством точек данных.

Глава 7. Работа с географическими картами и обогащение дашбордов при помощи языка разметки Markdown. В этой главе вы познакомитесь с еще одним распространенным типом визуализации. Существует множество способов отображения данных на карте. Мы рассмотрим два наиболее часто используемых: *точечный* (scatter map) и *картограмма* (choropleth map).

Глава 8. Определение частотности данных с помощью гистограмм и построение интерактивных таблиц. Эта глава посвящена разным способам построения гистограмм и их настройки, а также разделению данных различными методами с последующим подсчетом результирующих значений.

Глава 9. Машинное обучение: пусть данные говорят сами за себя. В этой главе вы узнаете о том, как работает кластеризация данных, и научитесь оценивать качество анализа. Также мы рассмотрим технику оценки кластеров и даже разработаем интерактивное приложение с реализацией кластеризации по методу *k*-средних.

Глава 10. Ускорение работы приложений с помощью улучшений функций обратного вызова. Здесь мы поговорим об использовании обратных вызовов на базе шаблонов с целью динамической модификации приложения на основе взаимодействия с пользователем и других факторов.

Глава 11. Ссылки и многостраничные приложения. В данной главе будет представлена новая архитектура, позволяющая создавать многостраничные приложения. Также мы рассмотрим технику использования ссылок в качестве элементов ввода или вывода со взаимодействием с другими элементами приложения.

Глава 12. Развертывание приложения. В этой главе мы обсудим вопросы развертывания созданного приложения на сервере с возможностью доступа к нему пользователям из любой точки мира. Здесь возможны разные варианты, и мы рассмотрим пару простых опций, которые могут оказаться полезными.

Глава 13. Следующие шаги. В заключительной главе книги мы поговорим о том, как можно вывести написанное приложение на новый уровень. Здесь мы дадим определенные рекомендации, советы и ресурсы, которые вам, возможно, захочется изучить самостоятельно.

Как извлечь максимум из книги

Для выполнения примеров из книги вам понадобится стабильное соединение с интернетом.

Если вы читаете книгу в формате PDF, мы рекомендуем вводить программный код вручную или использовать загруженный код из хранилища на GitHub (ссылка будет указана ниже). Это позволит вам избежать ошибок, связанных с копированием и вставкой текста.

Загрузите сопроводительные файлы

Сопроводительные файлы можно загрузить на странице книги на сайте издательства www.dmkpress.com.

Загрузите цветные изображения

По следующей ссылке вы можете скачать в виде PDF все рисунки и диаграммы, использованные в книге: https://static.packt-cdn.com/downloads/9781800568914_ColorImages.pdf.

Книга в видеофрагментах

Сопроводительные видеофрагменты к этой книге можно посмотреть по адресу <https://bit.ly/3vaXYQJ>.

Условные обозначения

На протяжении книги мы будем использовать следующие условные обозначения и шрифты.

Код в тексте: так в тексте книги мы будем обозначать код, имена таблиц баз данных, имена папок, файлов, расширения файлов, пути, ссылки, пользовательский ввод. Пример: «Наш набор данных будет состоять из файлов в папке `data`, находящейся в корне репозитория».

Блоки кода будут выделены следующим образом:

```
import plotly.express as px
gapminder = px.data.gapminder()
gapminder
```

Важные места в коде будут подсвечены жирным шрифтом, как показано ниже:

```
import os
import pandas as pd
pd.options.display.max_columns = None
os.listdir('data')
['PovStatsSeries.csv',
 'PovStatsCountry.csv',
 'PovStatsCountry-Series.csv',
 'PovStatsData.csv',
 'PovStatsFootNote.csv']
```

Жирным шрифтом также будут выделяться новые термины, важные слова и текст, который вы видите на экране. Например, таким образом будут обозначаться пункты меню. Пример:

«Еще одним важным столбцом является столбец **Limitations and exceptions**».

Советы или важные примечания

Будут выводиться так.

Часть I

Построение приложения на Dash

В этой вводной части вы познакомитесь с экосистемой фреймворка Dash и напишете свое первое простое приложение с минимальным функционалом.

Содержание этой части:

- глава 1 «Знакомство с экосистемой Dash»;
- глава 2 «Структура приложений Dash»;
- глава 3 «Работа с объектом Figure»;
- глава 4 «Подготовка и преобразование данных. Введение в Plotly Express».

Глава 1

Знакомство с экосистемой Dash

При работе с данными происходят постоянные изменения в объеме анализируемых данных, их источниках и типах. В связи с этим очень важно иметь возможность легко и просто комбинировать любые объемы данных из различных источников. Фреймворк *Dash* – это не только про исследование данных. Это про почти все стадии процесса анализа данных: от их поиска до создания полноценной рабочей среды.

В этой вводной главе мы познакомимся с экосистемой Dash и сконцентрируемся на внешнем макете приложения – той его части, с которой взаимодействует пользователь. Прочитав эту главу, вы сможете создать полностью работающее приложение с любыми визуальными элементами, но без интерактивных возможностей.

Темы, которые будут рассмотрены в главе:

- настройка окружения;
- исследование фреймворка Dash и сопутствующих пакетов;
- введение в базовую структуру приложения Dash;
- создание и запуск простого приложения Dash;
- добавление HTML и других компонентов в приложение;
- проектирование макета и управление темами.

Технические требования

В каждой главе будут применяться свои технические требования, но некоторые из них будут актуальны на протяжении всей книги.

Прежде всего у вас должен быть установлен *Python 3.6* или выше, который можно загрузить по адресу <https://www.python.org>. Также вам понадобится текстовый редактор или любая *интегрированная среда разработки* (integrated development environment – IDE) для написания и редактирования кода.

В этой главе мы будем использовать пакеты Dash, Dash HTML Components и Dash Bootstrap Components, которые можно загрузить вместе с остальными пакетами, следуя инструкции из следующего раздела. Исходный код и данные для этой книги можно скачать в репозитории GitHub по адресу <https://github.com/>

PacktPublishing/Interactive-Dashboards-and-Data-Apps-with-Plotly-and-Dash. Как я уже упомянул, в следующем разделе мы детально остановимся на настройке вашего рабочего окружения.

Исходный код к этой главе располагается в хранилище GitHub по адресу https://github.com/PacktPublishing/Interactive-Dashboards-and-Data-Apps-with-Plotly-and-Dash/tree/master/chapter_01.

Сопроводительные видеосюжеты к этой главе можно посмотреть по адресу <https://bit.ly/3atXPjc>.

Настройка окружения

Поскольку все пакеты, используемые в этой книге, стремительно развиваются и меняются, вы можете столкнуться с различиями в поведении ваших приложений. Чтобы в точности воспроизвести функционал приложений, заложенный при написании этой книги, мы рекомендуем вам клонировать репозиторий книги, установить версии пакетов, которые использовались при написании книги, и использовать в своих примерах приведенные наборы данных. Откройте командную строку, перейдите в папку, в которой хотите создать проект, и выполните следующие действия.

1. Создайте виртуальное окружение Python в папке с именем `dash_project` (или любой другой на ваше усмотрение). Это также приведет к созданию папки с выбранным именем:

```
python3 -m venv dash_project
```

2. Активируйте виртуальное окружение.
В Unix или macOS выполните следующую инструкцию:

```
source dash_project/bin/activate
```

Для Windows инструкция будет такой:

```
dash_project\Scripts\activate.bat
```

3. Перейдите в созданную папку:

```
cd dash_project
```

4. Клонировать репозиторий книги на GitHub:

```
git clone https://github.com/PacktPublishing/Interactive-Dashboards-and-Data-Apps-with-Plotly-and-Dash
```

5. В вашей папке должен появиться файл `requirements.txt` с перечислением всех необходимых пакетов и их версий. Вы можете установить все эти пакеты, перейдя в папку репозитория и выполнив команду `install`, как показано ниже:

```
cd Interactive-Dashboards-and-Data-Apps-with-Plotly-and-Dash/  
pip install -r requirements.txt
```

В папке `data` вы обнаружите копии наборов данных, которые были загружены с сайта <https://datacatalog.worldbank.org/dataset/poverty-and-equity-database>. Вы всегда можете загрузить свежие версии файлов, но здесь как с версиями пакетов – лучше использовать в точности те данные, которые применялись при написании книги, чтобы получать такие же результаты.

Для корректного отображения объектов и приложений Plotly в JupyterLab необходимо также установить Node.js по адресу <https://nodejs.org>.

Еще вам нужно установить расширение *JupyterLab Plotly*, что можно сделать, запустив следующую инструкцию из командной строки в вашем виртуальном окружении:

```
jupyter labextension install jupyterlab-plotly@4.14.1
```

Обратите внимание, что номер версии в конце строки должен совпадать с версией Plotly, которую вы используете. Вы можете изменить версию, не забыв при этом обновить и сам пакет Plotly.

Теперь вы полностью готовы двигаться дальше. В каждой следующей главе мы будем развивать идеи, озвученные в предыдущих главах, и дорабатывать созданные приложения, улучшая их функционал.

Основная цель – дать вам как можно больше возможностей для практики. Создать отдельный компонент Dash не составляет труда, но при объединении нескольких компонентов в приложении могут начаться сложности. Вы прочувствуете это, когда вам придется обновлять макет приложения и выполнять рефакторинг кода, концентрируясь на деталях, но в то же время не упуская из вида картину в целом.

Итак, окружение мы подготовили, пришло время рассказать о фреймворке Dash.

Исследование фреймворка Dash и сопутствующих пакетов

Хотя в этом нет строгой необходимости, все же вам полезно будет узнать, какие основные компоненты используются в *Dash*, чтобы уверенно чувствовать себя при разработке более сложных приложений и знать, куда обращаться за помощью.



Рис. 1.1. Из чего сделан Dash

Как видно на рис. 1.1, Dash использует фреймворк *Flask* на стороне сервера. Для построения диаграмм применяется графическая библиотека *Plotly* – это не строгое требование, но эта библиотека обладает самыми богатыми возможностями и поддержкой. Библиотека *React* используется для управления компонентами. По сути, любое приложение Dash можно воспринимать как одно-

страничное приложение React. Но гораздо важнее для нас сейчас узнать, какие пакеты используются в процессе создания приложения, и именно об этом мы поговорим далее.

Примечание

Одним из главных преимуществ фреймворка Dash является то, что он позволяет создавать полностью интерактивные приложения и интерфейсы для работы с данными и аналитикой с использованием чистого Python и без необходимости изучать HTML, CSS или JavaScript.

Совет

Тем, кто знаком с библиотекой Matplotlib, будет полезно узнать, что существуют инструменты для преобразования объектов Matplotlib в объекты Plotly. Таким образом, создав элемент в Matplotlib, вы сможете сконвертировать его в Plotly с помощью всего одной функции `mpl_to_plotly`. На момент написания книги такой функционал поддерживался только для Matplotlib версии не выше 3.0.3. Ниже приведен пример использования этой функции.

```
%config InlineBackend.figure_format = 'retina'  
import matplotlib.pyplot as plt  
from plotly.tools import mpl_to_plotly  
  
mpl_fig, ax = plt.subplots()  
ax.scatter(x=[1, 2, 3], y=[23, 12, 34])  
plotly_fig = mpl_to_plotly(mpl_fig)  
plotly_fig
```

Пакеты, содержащиеся во фреймворке Dash

Dash – это не один большой пакет, который содержит все необходимое. Это, скорее, собрание пакетов, каждый из которых служит конкретной цели. В дополнение, как мы увидим позже, существует большое количество сторонних пакетов, которые используются совместно с Dash, а сообщество создает собственные библиотеки для работы с этим фреймворком.

Ниже приведены основные пакеты, входящие в состав фреймворка Dash, которые мы будем изучать в этой главе:

- **Dash:** это базовый пакет, представляющий основу любого приложения посредством объекта `dash.Dash`. Также в этом пакете представлен функционал для управления интерактивностью и исключениями, что вы увидите при создании приложения;

- **Dash Core Components:** этот пакет содержит интерактивные компоненты, которыми управляет пользователь. Выпадающие списки, календари, ползунки и многое другое – это все есть в этом пакете. В главе 2 мы будем подробно говорить об этих компонентах применительно к интерактивности приложений, а во второй части книги еще более детально обсудим нюансы их использования;
- **Dash HTML Components:** в этом пакете представлены все возможные теги HTML в виде классов Python. Именно здесь происходит преобразование Python в HTML. К примеру, вы можете написать в Python `dash_html_components.H1('Hello, World')`, и этот код сгенерирует следующую разметку HTML: `<h1>Hello, World</h1>` – и соответствующим образом отобразит ее в браузере;
- **Dash Bootstrap Components:** это сторонний пакет, добавляющий фреймворку Dash функциональности Bootstrap. Этот пакет и его компоненты отвечают главным образом за макет и визуальное отображение элементов. Используя его, можно, например, разместить элементы приложения бок о бок или один над другим, определяя их размеры в зависимости от размера окна браузера, или настроить цветовую гамму приложения особым образом.

Совет

Чтобы установить все основные пакеты Dash, достаточно установить главный пакет. Это также позволит сохранить консистентность версий сопутствующих пакетов. Просто выполните инструкцию `pip install dash` из командной строки. Для обновления пакетов воспользуйтесь командой `pip install dash --upgrade`.

Теперь пришло время взглянуть на базовую структуру типичного приложения Dash.

Введение в базовую структуру приложения Dash

На рис. 1.2 условно показан процесс создания приложения Dash. К примеру, у нас есть файл с именем `app.py` (вы можете назвать его по своему усмотрению). Содержимое файла показано в правой колонке рисунка с условным разделением на секции, а в левой приведено описание секции.

Давайте рассмотрим каждый из приведенных этапов отдельно:

- **импорт (стандартная заготовка):** как и любой модуль Python, мы начинаем написание приложения с импортирования необходимых пакетов, давая им привычные псевдонимы¹;

¹ Здесь и далее: начиная с версии фреймворка Dash 2.0 следует использовать следующий синтаксис импорта: `from dash import html` и `from dash import dcc`, поскольку указанный синтаксис является устаревшим. – *Прим. перев.*

Части приложения	app.py
Импорт (стандартная заготовка)	<pre>import dash import dash_html_components as html import dash_core_components as dcc</pre>
Создание экземпляра приложения	<pre>app = dash.Dash(__name__)</pre>
Макет приложения: список HTML и/или интерактивных компонентов	<pre>app.layout = html.Div([dcc.Dropdown() dcc.Graph() ...])</pre>
Функции обратного вызова	<pre>@app.callback() ... @app.callback() ...</pre>
Запуск приложения	<pre>if __name__ == '__main__': app.run_server()</pre>

Рис. 1.2. Структура приложения Dash

- **создание экземпляра приложения:** простой способ создать приложение посредством инициализации переменной `app`. В качестве параметра передается значение `__name__`, чтобы Dash мог легко находить статические ресурсы, используемые в приложении;
- **макет приложения:** этому этапу мы посвятим большую часть данной главы. Именно здесь мы создаем все пользовательские элементы. Для этого мы обычно определяем контейнер (`html.Div`), принимающий в качестве аргумента `children` список дочерних компонентов. Эти компоненты будут последовательно отображаться при запуске приложения – один под другим. В следующем разделе мы создадим простейшее приложение с минималистическим макетом;
- **функции обратного вызова:** эту тему мы подробно начнем обсуждать в главе 2, посвященной интерактивности приложений. Пока вам достаточно будет знать, что здесь определяются функции для связывания визуальных элементов, в результате чего мы получаем функционал приложения. Обычно функции являются независимыми, они не должны быть объявлены внутри контейнера, а их порядок в модуле не имеет значения;
- **запуск приложения:** здесь мы осуществляем так называемый запуск приложения, если применить идиому о запуске модулей Python в качестве скриптов.

Итак, как я и обещал, мы уже готовы к написанию своих первых строк!

Создание и запуск простого приложения Dash

Держа в уме структуру приложения, которую мы только что обсудили, за исключением функций обратного вызова, давайте попробуем построить простейшее приложение.

Создайте файл `app.py` и введите в него следующий код.

1. Импорт необходимых пакетов с псевдонимами:

```
import dash
import dash_html_components as html
```

2. Создание экземпляра приложения:

```
app = dash.Dash(__name__)
```

3. Создание макета приложения:

```
app.layout = html.Div([
    html.H1('Hello, World!')
])
```

4. Запуск приложения:

```
if __name__ == '__main__':
    app.run_server(debug=True)
```

Позвольте сделать пару замечаний перед запуском приложения. Я настоятельно рекомендую не пользоваться копированием и вставкой кода. Пишите вручную. Вы должны запоминать вводимые конструкции. Кроме того, в процессе написания кода вам будут показываться подсказки в среде выполнения, и очень важно обращать внимание на предлагаемые возможности для компонентов, классов или функций.

Макет нашего приложения содержит единственный элемент, переданный как список элементу `html.Div` в качестве параметра `children`. Этот элемент будет преобразован в тег `H1`.

Заметьте также, что я передал на вход методу `app.run_server` параметр `debug` со значением `True`. Это позволяет активировать инструменты отладки, помогающие при разработке приложения.

Итак, мы готовы запустить наше первое приложение Dash. Для этого из командной строки, находясь в папке с файлом `app.py`, выполните следующую инструкцию:

```
python app.py
```

Если третья версия Python не установлена в вашей системе по умолчанию, вам может потребоваться указать номер версии вручную, как показано ниже:

```
python3 app.py
```

Вы увидите вывод, показанный на рис. 1.3, из которого следует, что приложение запущено.

```
> python app.py
Dash is running on http://127.0.0.1:8050/

* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
```

Рис. 1.3. Вывод командной строки после запуска приложения

Поздравляем! Ваше первое приложение Dash готово! Теперь, если вы введете в адресную строку браузера ссылку, указанную в выводе командной строки (<http://127.0.0.1:8050/>), то увидите строку *Hello, World!* в виде заголовка первого уровня (H1). Кроме того, в выводе командной строки указано, что в данный момент запущено серверное приложение Flask с именем `app`, а также присутствует предупреждение о том, что это сервер для разработки, а не для выпуска приложения. О разворачивании приложений мы будем говорить позже, сейчас же вам достаточно знать, что этого сервера нам вполне хватит для разработки и тестирования приложения. Вывод приложения показан на рис. 1.4.

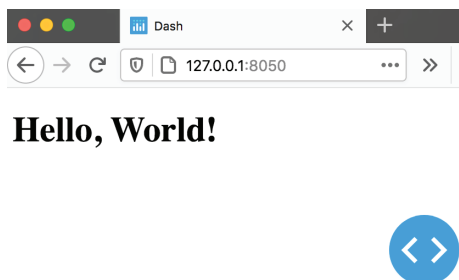


Рис. 1.4. Приложение в браузере

Под текстом, выведенным в виде заголовка первого уровня, вы можете заметить голубой кружок со стрелками. Щелкните по нему, и откроются дополнительные опции *отладки*, которые пригодятся вам при наличии в приложении функций обратного вызова и появлении ошибок. Если в качестве параметра при запуске приложения передать `debug=False`, эта кнопка отображаться не будет.

Теперь, когда вы создали простое приложение Dash и разобрались в базовой структуре макета, пришло время присмотреться к двум пакетам, которые используются для добавления и управления визуальными элементами. Первый пакет именуется **Dash HTML Components**, а второй – **Dash Bootstrap Components**.

Добавление HTML и других компонентов в приложение

С этого момента и до конца главы мы будем главным образом говорить об атрибуте нашего приложения `app.layout` и вносить в него изменения. Делать