

# Оглавление

Об авторах .....	14
О научном редакторе .....	15
Предисловие .....	16
Для кого написана эта книга .....	17
Обзор содержания книги .....	17
Как работать с книгой наиболее эффективно .....	19
Загрузка файлов с кодом примеров .....	19
Условные обозначения .....	20
От издательства .....	21
<b>Глава 1. Python сегодняшнего дня .....</b>	<b>22</b>
Где мы находимся и куда направляемся .....	23
Что делать с Python 2 .....	24
Как оставаться в курсе происходящего .....	26
Документы PEP .....	27
Активные сообщества .....	29
Другие ресурсы .....	31
Итоги .....	33
<b>Глава 2. Современные среды разработки для Python .....</b>	<b>34</b>
Технические требования .....	35
Экосистема пакетов Python .....	35
Установка пакетов Python с помощью pip .....	36

Изоляция среды выполнения . . . . .	38
Изоляция на уровне приложения и на уровне системы . . . . .	41
Изоляция сред на уровне приложений . . . . .	43
Poetry как система управления зависимостями . . . . .	45
Изоляция сред на уровне системы . . . . .	50
Контейнеризация и виртуализация . . . . .	52
Виртуальные среды с Docker . . . . .	54
Виртуальные среды разработки на базе Vagrant . . . . .	73
Популярные средства повышения производительности . . . . .	75
Специализированные оболочки Python . . . . .	76
IPython . . . . .	78
Как встроить оболочку в сценарии и программы . . . . .	80
Интерактивные отладчики . . . . .	82
Другие средства повышения производительности . . . . .	83
Итоги . . . . .	85
<b>Глава 3. Новые возможности Python . . . . .</b>	<b>86</b>
Технические требования . . . . .	87
Недавние добавления в Python . . . . .	87
Операторы слияния и обновления для словарей . . . . .	88
Выражения присваивания . . . . .	93
Аннотации обобщенных типов . . . . .	97
Чисто позиционные параметры . . . . .	99
Модуль zoneinfo . . . . .	101
Модуль graphlib . . . . .	103
Не самые свежие, но важные новшества . . . . .	107
Функция <code>breakpoint()</code> . . . . .	108
Режим разработки . . . . .	109
Функции <code>__getattr__()</code> и <code>__dir__()</code> на уровне модуля . . . . .	111
Форматирование строк с помощью f-строк . . . . .	112
Символы подчеркивания в числовых литералах . . . . .	114
Модуль <code>secrets</code> . . . . .	114

Чего стоит ожидать в будущем? . . . . .	116
Оператор   для объединения типов . . . . .	116
Структурное сопоставление с шаблоном . . . . .	117
Итоги . . . . .	122
<b>Глава 4. Python в сравнении с другими языками . . . . .</b>	<b>123</b>
Технические требования . . . . .	124
Модель классов и объектно-ориентированное программирование . . . . .	124
Доступ к надклассам . . . . .	126
Множественное наследование и порядок разрешения методов . . . . .	128
Инициализация экземпляров класса . . . . .	133
Паттерны обращения к атрибутам . . . . .	137
Дескрипторы . . . . .	138
Свойства . . . . .	145
Динамический полиморфизм . . . . .	150
Перегрузка операторов . . . . .	152
Перегрузка функций и методов . . . . .	159
Классы данных . . . . .	163
Функциональное программирование . . . . .	166
Лямбда-функции . . . . .	168
Функции map(), filter() и reduce() . . . . .	170
Частичные объекты и частичные функции . . . . .	173
Генераторы . . . . .	174
Генераторные выражения . . . . .	176
Декораторы . . . . .	176
Перечисления . . . . .	179
Итоги . . . . .	182
<b>Глава 5. Интерфейсы, паттерны и модульность . . . . .</b>	<b>183</b>
Технические требования . . . . .	184
Интерфейсы . . . . .	185
Немного истории: zope.interface . . . . .	187
Аннотации функций и абстрактные базовые классы . . . . .	195
Интерфейсы с аннотациями типов . . . . .	201

Инверсия управления и внедрение зависимостей .....	204
Инверсия управления в приложениях .....	206
Фреймворки внедрения зависимостей .....	214
Итоги .....	219
<b>Глава 6. Конкурентное выполнение .....</b>	<b>221</b>
Технические требования .....	222
Что такое конкурентное выполнение? .....	222
Многопоточность .....	225
Что такое многопоточность? .....	225
Как работают потоки в Python .....	229
Когда использовать многопоточность? .....	230
Пример многопоточного приложения .....	234
Многопроцессность .....	251
Встроенный модуль multiprocessing .....	254
Пулы процессов .....	258
Использование multiprocessing.dummy в качестве интерфейса многопоточности .....	260
Асинхронное программирование .....	261
Кооперативная многозадачность и асинхронный ввод/вывод .....	262
Ключевые слова async и await .....	263
Практический пример асинхронного программирования .....	267
Интеграция неасинхронного кода с async при помощи объектов Future ...	270
Итоги .....	274
<b>Глава 7. Событийно-ориентированное программирование .....</b>	<b>276</b>
Технические требования .....	277
Что такое событийное программирование? .....	277
Событийное != асинхронное .....	278
Событийное программирование в графическом интерфейсе пользователя .....	280
Событийное взаимодействие .....	282
Разные стили событийного программирования .....	284
Стиль с обратными вызовами .....	285

Стиль с субъектами .....	286
Стиль с топиками .....	291
Событийные архитектуры .....	293
Очереди событий и сообщений .....	294
Итоги .....	298
<b>Глава 8. Элементы метапрограммирования</b> .....	<b>299</b>
Технические требования .....	300
Что такое метапрограммирование? .....	300
Как использовать декораторы, чтобы изменить поведение функции перед вызовом .....	301
Следующий этап: декораторы классов .....	303
Вмешательство в процесс создания экземпляра класса .....	308
Метаклассы .....	311
Общий синтаксис .....	312
Использование метаклассов .....	316
Подводные камни метаклассов .....	319
Метод <code>__init_subclass__()</code> как альтернатива для метаклассов .....	320
Генерация кода .....	322
<code>exec</code> , <code>eval</code> и <code>compile</code> .....	323
Абстрактное синтаксическое дерево .....	324
Перехватчики импортирования .....	326
Примечательные примеры генерации кода в Python .....	327
Итоги .....	330
<b>Глава 9. Интеграция Python с C и C++</b> .....	<b>331</b>
Технические требования .....	333
C и C++ как основа расширяемости Python .....	333
Компиляция и загрузка расширений Python на C .....	334
Зачем нужны расширения .....	336
Улучшение быстродействия на критических участках кода .....	337
Интеграция существующего кода на других языках .....	338
Интеграция сторонних динамических библиотек .....	339
Создание эффективных структур данных .....	339

Как писать расширения . . . . .	340
Расширения на чистом С . . . . .	341
Разработка расширений на Cython . . . . .	359
Недостатки расширений . . . . .	366
Дополнительная сложность . . . . .	366
Более сложная отладка . . . . .	367
Взаимодействие с динамическими библиотеками без расширений . . . . .	368
Модуль ctypes . . . . .	369
CFFI . . . . .	376
Итоги . . . . .	377
<b>Глава 10. Автоматизация тестирования и контроля качества . . . . .</b>	<b>379</b>
Технические требования . . . . .	380
Принципы разработки через тестирование . . . . .	381
Написание тестов с помощью pytest . . . . .	384
Параметризация тестов . . . . .	391
Фикстуры pytest . . . . .	394
Суррогаты . . . . .	403
Mock-объекты и модуль unittest.mock . . . . .	406
Автоматизация контроля качества . . . . .	411
Тестовое покрытие . . . . .	411
Средства проверки стиля программирования и статического анализа кода . . . . .	416
Статический анализ типов . . . . .	420
Мутационное тестирование . . . . .	421
Полезные средства тестирования . . . . .	427
Моделирование реалистичных значений данных . . . . .	428
Моделирование значений времени . . . . .	429
Итоги . . . . .	430
<b>Глава 11. Упаковка и распространение кода Python . . . . .</b>	<b>432</b>
Технические требования . . . . .	433
Упаковка и распространение библиотек . . . . .	433
Как устроен пакет Python . . . . .	434

Дистрибутивы пакетов . . . . .	443
Регистрация и публикация пакетов . . . . .	448
Управление версиями пакетов и зависимостями . . . . .	451
Установка собственных пакетов . . . . .	455
Пакеты пространств имен . . . . .	457
Сценарии пакетов и точки входа . . . . .	459
Упаковка веб-приложений и веб-служб . . . . .	463
Манифест «12-факторное приложение» . . . . .	464
Docker для 12-факторных приложений . . . . .	466
Переменные окружения . . . . .	468
Роль переменных окружения во фреймворках приложений . . . . .	473
Создание автономных исполняемых файлов . . . . .	478
Когда стоит использовать автономные исполняемые файлы . . . . .	479
Популярные инструменты . . . . .	480
Безопасность кода Python в исполняемых пакетах . . . . .	487
Итоги . . . . .	489
<b>Глава 12. Наблюдение за поведением и быстродействием приложений . . . . .</b>	<b>490</b>
Технические требования . . . . .	491
Сбор информации об ошибках и журналирование . . . . .	491
Основы журналирования в Python . . . . .	492
Хорошие практики журналирования . . . . .	506
Распределенное журналирование . . . . .	509
Регистрация ошибок для последующего анализа . . . . .	511
Специализированные метрики приложения . . . . .	515
Prometheus . . . . .	518
Распределенная трассировка приложений . . . . .	527
Распределенная трассировка с помощью Jaeger . . . . .	531
Итоги . . . . .	537
<b>Глава 13. Оптимизация кода . . . . .</b>	<b>539</b>
Технические требования . . . . .	540
Типичные причины плохого быстродействия . . . . .	540

Сложность кода .....	541
Избыточное выделение ресурсов и утечки .....	545
Избыточный ввод/вывод и блокирующие операции .....	546
Профилирование кода .....	547
Профилирование загруженности процессора .....	549
Профилирование использования памяти .....	557
Уменьшение сложности за счет выбора структур данных .....	567
Поиск в списке .....	567
Использование множеств .....	568
Модуль collections .....	569
Архитектурные компромиссы .....	575
Эвристики и аппроксимирующие алгоритмы .....	575
Очереди задач и отложенная обработка .....	576
Вероятностные структуры данных .....	580
Кэширование .....	581
Итоги .....	590