

Оглавление

Предисловие	15
Благодарности	17
О книге	19
Для кого эта книга	19
Структура книги	20
О коде	21
Форум liveBook	22
Об авторе	24
Об обложке	25
От издательства	26

Часть I. Почему это важно?

Глава 1. Крупный план	28
1.1. Python — язык для корпоративных приложений	30
1.1.1. Времена меняются	30
1.1.2. Что мне нравится в Python	30
1.2. Python — язык для обучения	31

1.3. Дизайн — это процесс	32
1.3.1. Пользовательский опыт.....	33
1.3.2. Вы уже были здесь раньше	35
1.4. Дизайн обеспечивает высокое качество ПО.....	35
1.4.1. Соображения по дизайну ПО.....	36
1.4.2. Органически выращенное ПО	37
1.5. Когда нужно уделять время дизайну	40
1.6. Новые начинания.....	41
1.7. Дизайн демократичен	41
1.7.1. Хладнокровие.....	42
1.8. Как пользоваться этой книгой.....	44
Итоги.....	45

Часть II.

Основы проектирования

Глава 2. Разделение ответственности.....	48
2.1. Организация пространства имен.....	49
2.1.1. Пространства имен и инструкция <code>import</code>	50
2.1.2. Лики импортирования.....	53
2.1.3. Пространства имен предотвращают коллизии.....	55
2.2. Иерархия разделения в Python	57
2.2.1. Функции	57
2.2.2. Классы.....	64
2.2.3. Модули	71
2.2.4. Пакеты	72
Итоги.....	75
Глава 3. Абстракция и инкапсуляция.....	76
3.1. Что такое абстракция?	77
3.1.1. Черный ящик.....	77
3.1.2. Абстракция подобна луковице.....	79
3.1.3. Абстракция упрощает.....	82
3.1.4. Декомпозиция обеспечивает возможность абстракции.....	83

3.2. Инкапсуляция	84
3.2.1. Конструкции инкапсуляции в Python.....	84
3.2.2. Ожидания приватности в Python.....	86
3.3. Попробуйте сами	86
3.3.1. Рефакторинг	88
3.4. Стили программирования тоже являются абстракцией	90
3.4.1. Процедурное программирование	90
3.4.2. Функциональное программирование	91
3.4.3. Декларативное программирование.....	93
3.5. Типизация, наследование и полиморфизм	94
3.6. Распознавание неправильной абстракции.....	97
3.6.1. Как корове седло	97
3.6.2. Подходите ко всему с умом	98
Итоги	99
Глава 4. Создание дизайна для производительности	100
4.1. Сквозь время и пространство	101
4.1.1. Сложность немного... сложна	102
4.1.2. Временная сложность.....	103
4.1.3. Пространственная сложность.....	107
4.2. Производительность и типы данных	109
4.2.1. Типы данных для постоянного времени	110
4.2.2. Типы данных для линейного времени	111
4.2.3. Пространственная сложность операций на типах данных.....	111
4.3. Работоспособность, правильность и быстрота	115
4.3.1. Сделайте работоспособным	115
4.3.2. Сделайте правильным	116
4.3.3. Сделайте быстрым.....	120
4.4. Инструменты	121
4.4.1. timeit	121
4.4.2. Профилирование ЦП	123

4.5. Попробуйте сами.....	125
Итоги.....	126
Глава 5. Тестирование ПО	127
5.1. Что такое тестирование ПО?	128
5.1.1. Соответствует ли содержимое этикетке?	128
5.1.2. Суть функционального тестирования.....	129
5.2. Подходы к функциональному тестированию	131
5.2.1. Ручное тестирование	131
5.2.2. Автоматизированное тестирование.....	131
5.2.3. Приемочное тестирование.....	132
5.2.4. Юнит-тестирование	134
5.2.5. Интеграционное тестирование.....	136
5.2.6. Пирамида тестирования	137
5.2.7. Регрессионное тестирование	138
5.3. Констатация фактов.....	139
5.4. Юнит-тестирование с помощью unittest.....	140
5.4.1. Организация тестов с помощью unittest.....	140
5.4.2. Выполнение тестов с помощью unittest.....	141
5.4.3. Написание первого теста с помощью unittest.....	141
5.4.4. Написание первого интеграционного теста с помощью unittest.....	145
5.4.5. Тестовые дублиры	148
5.4.6. Попробуйте сами.....	150
5.4.7. Написание дополнительных тестов	152
5.5. Тестирование с помощью фреймворка pytest	153
5.5.1. Организация тестов с помощью фреймворка pytest	154
5.5.2. Конвертирование тестов unittest в pytest.....	155
5.6. За пределами функционального тестирования.....	156
5.6.1. Тестирование производительности	156
5.6.2. Нагрузочное тестирование	157

5.7. Разработка на основе тестов: пример	158
5.7.1. Это образ мышления.....	158
5.7.2. Это философия	159
Итоги	160

Часть III.
Организация крупных систем

Глава 6. Разделение ответственности на практике	162
6.1. Приложение командной строки для создания закладок	163
6.2. Обзор приложения Bagk	164
6.2.1. Выгоды от разделения: реприза	165
6.3. Первоначальное разделение ответственности	166
6.3.1. Слой постоянства данных.....	168
6.3.2. Слой бизнес-логики.....	181
6.3.3. Слой визуализации	187
Итоги	197
Глава 7. Расширяемость и гибкость	198
7.1. Что такое расширяемый код?	199
7.1.1. Добавление новых форм поведения.....	199
7.1.2. Изменение существующего поведения	202
7.1.3. Слабая сопряженность.....	204
7.2. Решения проблемы жесткости	206
7.2.1. Отпустить на свободу: инверсия управления.....	206
7.2.2. Дьявол в мелочах: опора на интерфейсы.....	210
7.2.3. Борьба с энтропией: принцип надежности	211
7.3. Упражнение на растяжку.....	212
Итоги	217
Глава 8. Правила (и исключения) наследования	218
8.1. Наследование раньше.....	219
8.1.1. Серебряная пуля	219
8.1.2. Трудности иерархий.....	219

8.2. Наследование сейчас.....	222
8.2.1. Зачем нужно наследование?	222
8.2.2. Подстановка.....	223
8.2.3. Идеальный для наследования вариант использования	225
8.3. Наследование в Python	228
8.3.1. Проверка типов.....	228
8.3.2. Обращение к суперклассу.....	230
8.3.3. Множественное наследование и порядок разрешения методов.....	230
8.3.4. Абстрактные базовые классы.....	235
8.4. Наследование и композиция в приложении Bark	238
8.4.1. Рефакторинг для использования абстрактного базового класса	238
8.4.2. Окончательная проверка работы с наследованием.....	240
Итоги.....	241
Глава 9. Поддержание компактности	242
9.1. Насколько большим должен быть класс/функция/модуль?	243
9.1.1. Физический размер.....	243
9.1.2. Ограниченная ответственность	244
9.1.3. Сложность кода	244
9.2. Разложение сложности	250
9.2.1. Извлечение конфигурационной информации.....	250
9.2.2. Извлечение функций.....	253
9.3. Декомпозиция классов.....	256
9.3.1. Сложность инициализации.....	256
9.3.2. Извлечение классов и переадресация вызовов	259
Итоги.....	264
Глава 10. Достижение слабой сопряженности	265
10.1. Определение сопряженности	266
10.1.1. Сопряженность.....	266

10.1.2. Тесная сопряженность	267
10.1.3. Слабая сопряженность	270
10.2. Распознавание сопряженности	274
10.2.1. Излишняя зависимость	274
10.2.2. Стрельба дробью	275
10.2.3. Дырявая абстракция	275
10.3. Сопряженность в приложении Vark	277
10.4. Решение проблемы сопряженности	280
10.4.1. Передача сообщений пользователям	281
10.4.2. Постоянство хранения закладок	284
10.4.3. Попробуйте сами	285
Итоги	289

**Часть IV.
Что дальше?**

Глава 11. Только вперед	292
11.1. Что теперь?	292
11.1.1. Разработайте план	293
11.1.2. Исполните план	295
11.1.3. Отслеживайте свой прогресс	297
11.2. Паттерны проектирования	299
11.2.1. Взлеты и падения паттернов в Python	301
11.2.2. С чего начать	302
11.3. Распределенные системы	302
11.3.1. Режимы сбоя в распределенных системах	303
11.3.2. Обращение к состоянию приложения	304
11.3.3. С чего начать	305
11.4. Погружение в Python	305
11.4.1. Питоновский стиль	305
11.4.2. Языковые средства являются паттернами	306
11.4.3. С чего начать	307

11.5. Что вы узнали.....	308
11.5.1. Путешествие туда и обратно: рассказ разработчика	308
11.5.2. Выход из системы.....	310
Итоги.....	310
Приложение. Установка языка Python.....	312
A.1. Какую версию Python использовать?	313
A.2. «Системный» Python	313
A.3. Установка других версий Python.....	314
A.3.1. Скачайте официальный Python.....	314
A.3.2. Скачать с помощью Anaconda.....	316
A.4. Проверка установки	316