Оглавление

Предисловие **14**

Благодарности **16**

О книге **18**

Для кого эта книга 18

Структура книги 18

О коде в книге 19

Форум liveBook 20

Об авторе **21**

Иллюстрация на обложке **22**

От издательства **23**

Глава 1. На улицы! **24**

1.1. Что важно на улицах 25

1.2. Кто такой уличный кодер? 26

1.3. Великие уличные кодеры 28

1.3.1. Любознательность 29

1.3.2. Нацеленность на результат 29

1.3.3. Высокая производительность 30

1.3.4. Умение справляться со сложностями и неоднозначностями 30

1.4. Проблемы современной разработки ... 31
1.4.1. Слишком много технологий ... 33
1.4.2. Парапланеризм на парадигмах ... 34
1.4.3. Черные ящики технологий ... 35
1.4.4. Недооценка накладных расходов ... 36
1.4.5. Не моя работа ... 37
1.4.6. Рутина — это гениально ... 37
1.5. Чего нет в этой книге ... 38
1.6. Основные темы книги ... 38
Итоги ... 39

Глава 2. Практическая теория ... 40
2.1. Краткий обзор алгоритмов ... 41
2.1.1. «О-большое» должно быть приемлемым ... 44
2.2. Структуры данных изнутри ... 46
2.2.1. Строки ... 48
2.2.2. Массив ... 51
2.2.3. Список ... 52
2.2.4. Связанный список ... 53
2.2.5. Очередь ... 55
2.2.6. Словарь ... 55
2.2.7. Хеш-множества ... 59
2.2.8. Стек ... 59
2.2.9. Стек вызовов ... 60
2.3. К чему весь этот ажиотаж с типами? ... 61
2.3.1. Сила типов ... 62
2.3.2. Проверка правильности ... 63
2.3.3. Используйте фреймворк с умом ... 69
2.3.4. Типы вместо опечаток ... 73

2.3.5. Быть nullable или non-nullable? ... 75
2.3.6. Высокая производительность бесплатно ... 82
2.3.7. Ссылочные типы и типы значений ... 83
Итоги ... 87

Глава 3. Полезные антипаттерны ... 89
3.1. Если не сломано, сломай ... 90
3.1.1. Лицом к лицу с жестью ... 91
3.1.2. Ломайте скорее ... 92
3.1.3. Соблюдайте границы ... 93
3.1.4. Выделение общей функциональности ... 94
3.1.5. Пример веб-страницы ... 96
3.1.6. Не оставляйте за собой долгов ... 97
3.2. Пишите с нуля ... 98
3.2.1. Стирайте и переписывайте ... 99
3.3. Чините, даже если ничего не сломано ... 100
3.3.1. Гонка за будущим ... 100
3.3.2. Качество кода и культура поведения ... 102
3.4. Не бойтесь повторяться ... 104
3.4.1. Повторное использование или копирование? ... 109
3.5. Изобретайте ... 111
3.6. Не используйте наследование ... 114
3.7. Не используйте классы ... 117
3.7.1. Enum — это ням! ... 117
3.7.2. Структуры рулят! ... 119
3.8. Пишите плохой код ... 125
3.8.1. Не используйте If/Else ... 125
3.8.2. Используйте goto ... 127
3.9. Не пишите комментарии к коду ... 131

3.9.1. Подбирайте длинные имена ... 133
3.9.2. Эффективно используйте функции ... 134
Итоги ... 136

Глава 4. Распробуйте тестирование ... 138
4.1. Типы тестов ... 139
4.1.1. Ручное тестирование ... 139
4.1.2. Автоматизированное тестирование ... 139
4.1.3. Опасная жизнь: тестирование в рабочей среде ... 141
4.1.4. Выбор правильной методологии тестирования ... 142
4.2. Как перестать беспокоиться и полюбить тесты ... 144
4.3. Не используйте TDD и другие сокращения ... 152
4.4. Пишите тесты для своего же блага ... 153
4.5. Как понять, что именно тестировать ... 154
4.5.1. Уважайте границы ... 155
4.5.2. Покрытие кода ... 157
4.6. Не пишите тесты ... 160
4.6.1. Не пишите код ... 160
4.6.2. Ограничьтесь выборочными тестами ... 160
4.7. Пусть тестированием займется компилятор ... 161
4.7.1. Как исключить проверки на null ... 161
4.7.2. Как исключить проверки диапазона ... 165
4.7.3. Как исключить проверки допустимых значений ... 167
4.8. Именование тестов ... 170
Итоги ... 171

Глава 5. Вознаграждение за рефакторинг ... 172
5.1. Зачем нужен рефакторинг? ... 173
5.2. Изменения архитектуры ... 174
5.2.1. Выделение компонентов ... 177
5.2.2. Оценка объема работы и риска ... 178

5.2.3. Престиж ... 179
5.2.4. Рефакторинг, чтобы упростить рефакторинг ... 181
5.2.5. Финальное усилие ... 188
5.3. Надежный рефакторинг ... 189
5.4. Когда рефакторинг не нужен ... 191
Итоги ... 192

Глава 6. Все внимание безопасности ... 193
6.1. Что еще, кроме хакеров ... 194
6.2. Моделирование угроз ... 196
6.2.1. Модели угроз карманного формата ... 198
6.3. Написание безопасных веб-приложений ... 200
6.3.1. Проектирование с учетом требований безопасности ... 201
6.3.2. Повышение безопасности через неясность ... 202
6.3.3. Не используйте собственные механизмы безопасности ... 203
6.3.4. Атаки путем внедрения SQL-кода ... 204
6.3.5. Межсайтовый скриптинг ... 211
6.3.6. Межсайтовая подделка запроса (CSRF) ... 217
6.4. Флуд ... 219
6.4.1. Не используйте капчу ... 219
6.4.2. Альтернативы капче ... 220
6.4.3. Не применяйте кэш ... 221
6.5. Хранение секретов ... 222
6.5.1. Хранение секретов в исходном коде ... 222
Итоги ... 230

Глава 7. Самостоятельная оптимизация ... 231
7.1. Решаем правильную проблему ... 232
7.1.1. Простой бенчмаркинг ... 233
7.1.2. Производительность и время отклика ... 236

7.2. Анатомия медлительности ... 238
7.3. Начните сверху ... 239
7.3.1. Вложенные циклы ... 241
7.3.2. Строко-ориентированное программирование ... 243
7.3.3. Вычисление 2b ‖ !2b ... 244
7.4. Разбиваем бутылку по горлышку ... 245
7.4.1. Не упаковывайте данные ... 246
7.4.2. Производите вычисления локально ... 247
7.4.3. Разделяйте зависимые процессы ... 248
7.4.4. Будьте предсказуемы ... 250
7.4.5. SIMD ... 252
7.5. Ввод и вывод ... 255
7.5.1. Ускоряйте ввод/вывод ... 255
7.5.2. Делайте ввод/вывод неблокирующим ... 257
7.5.3. Архаичные способы ... 259
7.5.4. Современные операторы async/await ... 260
7.5.5. Подводные камни асинхронного ввода/вывода ... 261
7.6. Если ничего не помогает, кэшируйте ... 262
Итоги ... 262

Глава 8. Приятная масштабируемость ... 264
8.1. Не используйте блокировки ... 266
8.1.1. Блокировка с двойной проверкой ... 274
8.2. Смиритесь с несоответствиями ... 277
8.2.1. Страшный NOLOCK ... 278
8.3. Не кэшируйте подключения к базе данных ... 280
8.3.1. В виде ORM ... 284
8.4. Не используйте потоки ... 285
8.4.1. Подводные камни асинхронного кода ... 289
8.4.2. Многопоточность и асинхронность ... 290

8.5. Уважайте монолит 291
Итоги 292

Глава 9. Жизнь с ошибками 294
9.1. Не исправляйте ошибки 296
9.2. Ужас ошибок 297
9.2.1. Неприятная правда об исключениях 298
9.2.2. Не перехватывайте исключения 300
9.2.3. Устойчивость к исключениям 303
9.2.4. Устойчивость без транзакций 308
9.2.5. Исключения и ошибки 309
9.3. Не занимайтесь отладкой 311
9.3.1. Отладка printf() 312
9.3.2. Дамп-дайвинг 313
9.3.3. Продвинутая отладка с помощью резиновой уточки 317
Итоги 318