

Оглавление

Предисловие	13
О чем эта книга	14
Как читать эту книгу	15
Использование примеров кода	15
Как с нами связаться	15
Благодарности	16
Об авторах	17
От издательства	18
Глава 1. Код должен быть простым для понимания	19
Что делает код «лучше»?	20
Фундаментальная теорема читаемости	21
Меньше — значит лучше?	21
Противоречит ли время-для-понимания другим целям?	22
Самое сложное	22

Часть I. Поверхностные улучшения

Глава 2. Помещаем в имена полезную информацию	25
Выбираем конкретные слова	26
Избегаем общих имен, например таких, как <code>tmp</code> и <code>retval</code>	28
Используйте конкретные имена вместо абстрактных	31
Добавление дополнительной информации к имени	33
Насколько длинным должно быть имя?	36
Использование форматирования имен для передачи их смысла	38
Итог	40
Глава 3. Имена, которые нельзя понять неправильно	41
Пример: <code>Filter()</code> (фильтрация)	42
Пример: <code>Clip(text, lenght)</code> (обрезать)	42

Применяйте префиксы min и max для (включающих) границ	43
Используйте в именах границ слова first и last	44
Используйте имена begin и end для включающе-исключающих границ	44
Называем булевы переменные	45
Оправдываем ожидания пользователей	45
Пример: оценка нескольких вариантов названия	47
Итог	49
Глава 4. Эстетичность.	51
Почему красота имеет значение?	52
Перераспределение разрывов строк сделает код более последовательным и компактным	53
Избавляемся от неоднородности с помощью методов	55
Выравнивание столбцов.	57
Выберите определенный порядок и придерживайтесь его	58
Объединяем объявления в блоки	58
Разбиваем код на абзацы.	59
Персональный стиль или единообразие?	61
Итог	62
Глава 5. Комментируем мудро	63
Что НЕ нужно комментировать.	64
Записываем ваши мысли	67
Поставьте себя на место читателя	69
Преодоление «творческого кризиса»	74
Итог	75
Глава 6. Комментарии должны быть четкими и компактными.	77
Старайтесь комментировать компактно	78
Избегайте двусмысленных местоимений и указательных слов.	78
«Полируем» нечеткие предложения.	79
Четко описываем поведение функции	79
Используйте примеры ввода/вывода, иллюстрирующие спорные ситуации	80
Описывайте цели вашего кода.	81
Комментарии, содержащие названия параметров функций.	82
Употребляйте максимально содержательные слова	83
Итог	83

Часть II. Упрощение цикла и логики

Глава 7. Как сделать поток команд управления

удобочитаемым	87
Порядок аргументов в условных конструкциях	88
Порядок блоков if/else	89
Условная конструкция ?: (также известная как тернарный оператор)	91
Избегайте циклов do/while	92
Слишком быстрый возврат из функции.	94
Пресловутый goto	94
Сокращаем количество вложенного кода	95
Можете ли вы отследить порядок выполнения вашей программы?	98
Итог	99

Глава 8. Разбиваем длинные выражения 101

Поясняющие переменные	102
Итоговые переменные	102
Используем законы де Моргана	103
Злоупотребление упрощенной логикой	104
Пример: боремся со сложной логикой	104
Разбиваем огромные утверждения	107
Еще один творческий способ упрощения выражений	108
Итог	109

Глава 9. Переменные и читаемость 111

Избавляемся от переменных	112
Сокращаем область видимости ваших переменных.	115
Используйте переменные, меняющие свое значение однократно	121
Последний пример	122
Итог	124

Часть III. Реорганизация кода

Глава 10. Выделяем побочные подзадачи 127

Вводный пример: findClosestLocation()	128
Чистый вспомогательный код.	130

Прочий универсальный код	130
Создавайте больше универсального кода	132
Функциональность, специфичная для проекта	133
Упрощаем существующий интерфейс	134
Изменяем интерфейс под собственные нужды	135
Все хорошо в меру	136
Итог	137
Глава 11. Одна задача в любой момент времени	139
Задания могут быть маленькими	141
Извлекаем значения из объекта	142
Более объемный пример	146
Итог	149
Глава 12. Превращаем мысли в код	151
Четко описываем логику	152
Библиотеки нам помогут	153
Применяем этот метод к более объемным задачам	154
Описание решения задачи на русском языке	156
Итог	158
Глава 13. Пишите меньше кода	161
Не беспокойтесь о реализации этой функции — она вам не понадобится	162
Критикуйте и разделяйте ваши требования	162
Сохраняйте базу кода небольшой	164
Старайтесь изучать возможности доступных библиотек	165
Пример: использование инструментов UNIX вместо написания кода	166
Итог	167

Часть IV. Избранные темы

Глава 14. Тестирование и читаемость	171
Создавайте тесты, которые легко читать и обслуживать	172
Что не так с этим тестом?	172
Приводим тест в читаемый вид	173

Создание минимального тестового выражения	174
Реализуем пользовательские мини-языки.	175
Что было не так с тем тестом?	182
Разработка, ориентированная на тестирование	183
Не увлекайтесь!	185
Итог	186
Глава 15. Разработка и реализация счетчика минут и часов . . .	187
Постановка задачи.	188
Определение интерфейса класса	188
Первый подход: простое решение	191
Вторая попытка: реализация конвейерного дизайна.	194
Третья попытка: дизайн, при котором время делится на блоки.	197
Сравнение трех решений	202
Итог	202