

Роберт Мартин

ЧИСТЫЙ КОД

**Создание,
анализ
и рефакторинг**

- Что такое «чистый код»?
- Как улучшить плохой код?
- Почему чистый код часто «портится»?
- Почему в написании кода так важны мелочи?

Robert C. Martin

Clean Code:

A Handbook of Agile Software Craftsmanship

 PRENTICE
HALL



БИБЛИОТЕКА ПРОГРАММИСТА

Роберт Мартин

ЧИСТЫЙ КОД

Создание, анализ и рефакторинг



0114682

НТБ ТРТУ

347922, Россия, Ростовская обл.,
г. Таганрог, ул. Черокев 22
тел. (8634) 3719-80
e-mail: ntbt@yandex.ru



Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Ростов-на-Дону · Екатеринбург · Самара · Новосибирск
Киев · Харьков · Минск

2015

Роберт Мартин

**Чистый код: создание, анализ и рефакторинг.
Библиотека программиста**

Перевел с английского Е. Матвеев

Заведующий редакцией
Руководитель проекта
Ведущий редактор
Художественный редактор
Корректор
Верстка

А. Кривоцов
А. Юрченко
Ю. Сергиенко
Л. Адуевская
В. Листова
Е. Егорова

БИБЛИОТЕКА
Научно-техническое
отделение ЗНБ ЮФУ
(г. Таганрог)

7543721

Мартин Р.

M29 Чистый код: создание, анализ и рефакторинг. Библиотека программиста. — СПб.: Питер, 2015. — 464 с.: ил. — (Серия «Библиотека программиста»).

ISBN 978-5-496-00487-9

Даже плохой программный код может работать. Однако если код не является «чистым», это всегда будет мешать развитию проекта и компании-разработчика, отнимая значительные ресурсы на его поддержку и «укрошение».

Эта книга посвящена хорошему программированию. Она полна реальных примеров кода. Мы будем рассматривать код с различных направлений: сверху вниз, снизу вверх и даже изнутри. Прочитав книгу, вы узнаете много нового о коде. Более того, вы научитесь отличать хороший код от плохого. Вы узнаете, как писать хороший код и как преобразовать плохой код в хороший.

Книга состоит из трех частей. В первой части излагаются принципы, паттерны и приемы написания чистого кода; приводится большой объем примеров кода. Вторая часть состоит из практических сценариев нарастающей сложности. Каждый сценарий представляет собой упражнение по чистке кода или преобразованию проблемного кода в код с меньшим количеством проблем. Третья часть книги — концентрированное выражение ее сути. Она состоит из одной главы с перечнем эвристических правил и «запахов кода», собранных во время анализа. Эта часть представляет собой базу знаний, описывающую наш путь мышления в процессе чтения, написания и чистки кода.

12+ (Для детей старше 12 лет. В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ISBN 978-0132350884 (англ.)

ISBN 978-5-496-00487-9

© Prentice Hall, Inc.

© Перевод на русский язык ООО Издательство «Питер», 2015

© Издание на русском языке, оформление
ООО Издательство «Питер», 2015

Права на издание получены по соглашению с Prentice Hall, Inc. Upper Saddle River, New Jersey 07458.

Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

Информация, содержащаяся в данной книге, получена из источников, рассматриваемых издательством как надежные. Тем не менее, имея в виду возможные человеческие или технические ошибки, издательство не может гарантировать абсолютную точность и полноту приводимых сведений и не несет ответственности за возможные ошибки, связанные с использованием книги.

ООО «Питер Пресс», 192102, Санкт-Петербург, ул. Андреевская (д. Волкова), д. 3, литер А, пом. 7Н.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014,

58.11.12 — Книги печатные профессиональные, технические и научные.

Подписано в печать 17.09.14. Формат 70х100/16. Усл. п. л. 37,410. Доп. тираж. Заказ № 91.

Отпечатано в полном соответствии с качеством предоставленных издательством материалов
в ГП ПО «Псков-Полиграф». 180004, г. Псков, ул. Ротная, 34.

Содержание

Предисловие	14
Введение	20
Глава 1. Чистый код	23
Да будет код	24
Плохой код	25
Расплата за хаос	26
Грандиозная переработка	26
Отношение	27
Основной парадокс	28
Искусство чистого кода?	28
Что такое «чистый код»?	29
Мы — авторы	36
Правило бойскаута	37
Предыстория и принципы	37
Заключение	38
Литература	38
Глава 2. Содержательные имена (Тим Оттингер)	39
Имена должны передавать намерения программиста	40
Избегайте дезинформации	41
Используйте осмысленные различия	42
Используйте удобопроизносимые имена	44
Выбирайте имена, удобные для поиска	45
Избегайте схем кодирования имен	45
Венгерская запись	46
Префиксы членов классов	46
Интерфейсы и реализации	47
Избегайте мысленных преобразований	47
Имена классов	48
Имена методов	48
Избегайте остроумия	48
Выберите одно слово для каждой концепции	49
Воздержитесь от каламбуров	49
Используйте имена из пространства решения	50
Используйте имена из пространства задачи	50

Добавьте содержательный контекст	51
Не добавляйте избыточный контекст	53
Несколько слов напоследок	53
Глава 3. Функции	55
Компактность!	58
Блоки и отступы	59
Правило одной операции	59
Секции в функциях	60
Один уровень абстракции на функцию	61
Чтение кода сверху вниз: правило понижения	61
Команды switch	62
Используйте содержательные имена	64
Аргументы функций	64
Стандартные унарные формы	65
Аргументы-флаги	66
Бинарные функции	66
Тернарные функции	67
Объекты как аргументы	68
Списки аргументов	68
Глаголы и ключевые слова	68
Избавьтесь от побочных эффектов	69
Выходные аргументы	70
Разделение команд и запросов	70
Используйте исключения вместо возвращения кодов ошибок	71
Изолируйте блоки try/catch	72
Обработка ошибок как одна операция	72
Магнит зависимостей Error.java	73
Не повторяйтесь	73
Структурное программирование	74
Как научиться писать такие функции?	74
Завершение	75
Литература	78
Глава 4. Комментарии	79
Комментарии не компенсируют плохого кода	81
Объясните свои намерения в коде	81
Хорошие комментарии	81
Юридические комментарии	82
Информативные комментарии	82
Представление намерений	82
Прояснение	83
Предупреждения о последствиях	84
Комментарии TODO	85
Усиление	85
Комментарии Javadoc в общедоступных API	86
Плохие комментарии	86
Бормотание	86
Избыточные комментарии	87
Недостоверные комментарии	89

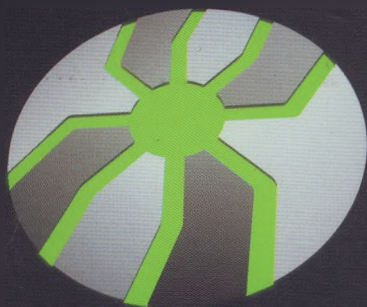
Обязательные комментарии	90
Журнальные комментарии	90
Шум	91
Опасный шум	93
Не используйте комментарии там, где можно использовать функцию или переменную	93
Позиционные маркеры	94
Комментарии за закрывающей фигурной скобкой	94
Ссылки на авторов	95
Закомментированный код	95
Комментарии HTML	96
Нелокальная информация	96
Слишком много информации	97
Неочевидные комментарии	97
Заголовки функций	97
Заголовки Javadoc во внутреннем коде	98
Пример	98
Литература	101
Глава 5. Форматирование	102
Цель форматирования	103
Вертикальное форматирование	103
Газетная метафора	104
Вертикальное разделение концепций	105
Вертикальное сжатие	106
Вертикальные расстояния	107
Вертикальное упорядочение	112
Горизонтальное форматирование	112
Горизонтальное разделение и сжатие	113
Горизонтальное выравнивание	114
Отступы	116
Вырожденные области видимости	117
Правила форматирования в группах	118
Правила форматирования от дядюшки Боба	118
Глава 6. Объекты и структуры данных	121
Абстракция данных	121
Антисимметрия данных/объектов	123
Закон Деметры	126
Крушение поезда	126
Гибриды	127
Скрытие структуры	127
Объекты передачи данных	128
Активные записи	129
Заключение	130
Литература	130
Глава 7. Обработка ошибок (Майк Физерс)	131
Используйте исключения вместо кодов ошибок	132
Начните с написания команды try-catch-finally	133

Используйте непроверяемые исключения	135
Передавайте контекст с исключениями	136
Определяйте классы исключений в контексте потребностей вызывающей стороны	136
Определите нормальный путь выполнения	138
Не возвращайте null	139
Не передавайте null	140
Заключение	141
Литература	141
Глава 8. Границы (Джеймс Гренинг)	142
Использование стороннего кода	143
Исследование и анализ границ	145
Изучение log4j	145
Учебные тесты: выгоднее, чем бесплатно	147
Использование несуществующего кода	148
Чистые границы	149
Литература	149
Глава 9. Модульные тесты	150
Три закона TTD	151
О чистоте тестов	152
Тесты как средство обеспечения изменений	153
Чистые тесты	154
Предметно-ориентированный язык тестирования	157
Двойной стандарт	157
Одна проверка на тест	159
Одна концепция на тест	161
F.I.R.S.T.	162
Заключение	163
Литература	163
Глава 10. Классы (совместно с Джеффом Лангром)	164
Строение класса	164
Инкапсуляция	165
Классы должны быть компактными!	165
Принцип единой ответственности (SRP)	167
Связность	169
Поддержание связности приводит к уменьшению классов	170
Структурирование с учетом изменений	176
Изоляция изменений	179
Литература	180
Глава 11. Системы (Кевин Дин Уомплер)	181
Как бы вы строили город?	182
Отделение конструирования системы от ее использования	182
Отделение main	184
Фабрики	184
Внедрение зависимостей	185

Применяйте инструментовку кода для повышения вероятности сбоев	220
Ручная инструментовка	221
Автоматизированная инструментовка	222
Заключение	223
Литература	224
Глава 14. Последовательное очищение	225
Реализация Args	226
Как я это сделал?	233
Args: черновик	233
На этом я остановился	245
О постепенном усовершенствовании	246
Аргументы String	248
Заключение	286
Глава 15. Внутреннее строение JUnit	287
Инфраструктура JUnit	288
Заключение	302
Глава 16. Переработка SerialDate	303
Прежде всего — заставить работать	304
...Потом очистить код	306
Заключение	320
Литература	321
Глава 17. Запахи и эвристические правила	322
Комментарии	323
C1: Неуместная информация	323
C2: Устаревший комментарий	323
C3: Избыточный комментарий	323
C4: Плохо написанный комментарий	323
C5: Закомментированный код	324
Рабочая среда	324
E1: Построение состоит из нескольких этапов	324
E2: Тестирование состоит из нескольких этапов	324
Функции	325
F1: Слишком много аргументов	325
F2: Выходные аргументы	325
F3: Флаги в аргументах	325
F4: Мертвые функции	325
Разное	325
G1: Несколько языков в одном исходном файле	325
G2: Очевидное поведение не реализовано	326
G3: Некорректное граничное поведение	326
G4: Отключенные средства безопасности	326
G5: Дублирование	327
G6: Код на неверном уровне абстракции	328
G7: Базовые классы, зависящие от производных	329

G8: Слишком много информации	329
G9: Мертвый код	330
G10: Вертикальное разделение	330
G11: Непоследовательность	330
G12: Балласт	331
G13: Искусственные привязки	331
G14: Функциональная зависть	331
G15: Аргументы-селекторы	332
G16: Непонятные намерения	333
G17: Неверное размещение	334
G18: Неуместные статические методы	334
G19: Используйте пояснительные переменные	335
G20: Имена функций должны описывать выполняемую операцию	335
G21: Понимание алгоритма	336
G22: Преобразование логических зависимостей в физические	336
G23: Используйте полиморфизм вместо if/Else или switch/Case	338
G24: Соблюдайте стандартные конвенции	338
G25: Заменяйте «волшебные числа» именованными константами	339
G26: Будьте точны	340
G27: Структура важнее конвенций	340
G28: Инкапсулируйте условные конструкции	341
G29: Избегайте отрицательных условий	341
G30: Функции должны выполнять одну операцию	341
G31: Скрытые временные привязки	342
G32: Структура кода должна быть обоснована	343
G33: Инкапсулируйте граничные условия	343
G34: Функции должны быть написаны на одном уровне абстракции	344
G35: Храните конфигурационные данные на высоких уровнях	345
G36: Избегайте транзитивных обращений	346
Java	347
J1: Используйте обобщенные директивы импорта	347
J2: Не наследуйте от констант	347
J3: Константы против перечислений	348
Имена	349
N1: Используйте содержательные имена	349
N2: Выбирайте имена на подходящем уровне абстракции	351
N3: По возможности используйте стандартную номенклатуру	352
N4: Недвусмысленные имена	352
N5: Используйте длинные имена для длинных областей видимости	353
N6: Избегайте кодирования	353
N7: Имена должны описывать побочные эффекты	354
Тесты	354
T1: Нехватка тестов	354
T2: Используйте средства анализа покрытия кода	354
T3: Не пропускайте тривиальные тесты	354
T4: Отключенный тест как вопрос	355
T5: Тестируйте граничные условия	355
T6: Тщательно тестируйте код рядом с ошибками	355

T7: Закономерности сбоев часто несут полезную информацию	355
T8: Закономерности покрытия кода часто несут полезную информацию	355
T9: Тесты должны работать быстро	356
Заключение	356
Литература	356
Приложение А. Многопоточность II	357
Пример приложения «клиент/сервер»	357
Найдите свои библиотеки	367
Зависимости между методами могут нарушить работу многопоточного кода	370
Повышение производительности	375
Взаимная блокировка	377
Тестирование многопоточного кода	381
Средства тестирования многопоточного кода	384
Полные примеры кода	385
Приложение Б. org.jfree.date.SerialDate	390
Приложение В. Перекрестные ссылки	455
Эпилог	458
Алфавитный указатель	459



SALD

САНКТ-ПЕТЕРБУРГСКАЯ
АНТИВИРУСНАЯ
ЛАБОРАТОРИЯ
ДАНИЛОВА

www.SALD.ru
8 (812) 336-3739

АНТИВИРУСНЫЕ
ПРОГРАММНЫЕ ПРОДУКТЫ

Вы читаете эту книгу по двум причинам. Во-первых, вы программист. Во-вторых, вы хотите повысить свою квалификацию. Отлично. Хороших программистов не хватает. Эта книга посвящена хорошему программированию. Она полна реальных примеров кода. Мы будем рассматривать код с различных направлений: сверху вниз, снизу вверх и даже изнутри. Прочитав книгу, вы узнаете много нового о коде. Более того, вы научитесь отличать хороший код от плохого. Вы узнаете, как писать хороший код и как преобразовать плохой код в хороший.

Книга состоит из трех частей. В первой части излагаются принципы, паттерны и приемы написания чистого кода; приводится большой объем примеров кода. Вторая часть состоит из практических сценариев нарастающей сложности. Каждый сценарий представляет собой упражнение по чистке кода или преобразованию проблемного кода в код с меньшим количеством проблем. Третья часть книги — концентрированное выражение ее сути. Она состоит из одной главы с перечнем эвристических правил и «запахов кода», собранных во время анализа. Эта часть представляет собой базу знаний, описывающую наш путь мышления в процессе чтения, написания и чистки кода.

Тема: Программирование. Тестирование и отладка ПО

Уровень пользователя: опытный

ISBN: 978-5-496-00487-9



9 785496 004879

ПИТЕР®

Заказ книг:

197198, Санкт-Петербург, а/я 127
тел.: (812) 703-73-74, postbook@piter.com

61093, Харьков-93, а/я 9130
тел.: (057) 758-41-45, 751-10-02, piter@kharkov.piter.com

www.piter.com — вся информация о книгах и веб-магазин