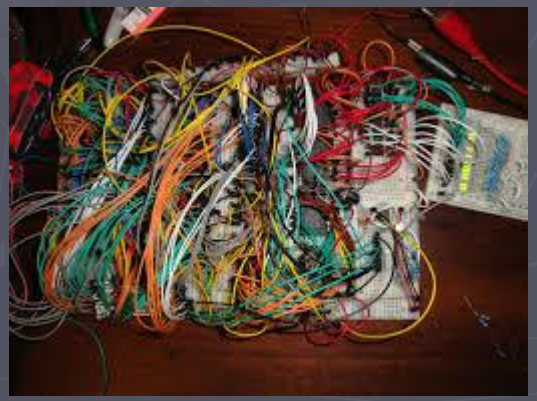
Антипаттерны проектирования. Чистый код.

Анти паттерны

- Spaggeticode

Спагетти-код — слабо структурированная и плохо спроектированная система, запутанная и очень сложная для понимания.



```
C     A weird program for calculating Pi written in Fortran.
C     From: Fink, D.G., Computers and the Human Mind, Anchor Books, 1966.

      PROGRAM PI
      DIMENSION TERM(100)
      N=1
3     TERM(N)=((-1)**(N+1))*(4./(2.*N-1.))
      N=N+1
      IF (N-101) 3,6,6
6     N=1
7     SUM98 = SUM98+TERM(N)
      WRITE(*,28) N, TERM(N)
      N=N+1
      IF (N-99) 7, 11, 11
11    SUM99=SUM98+TERM(N)
      SUM100=SUM99+TERM(N+1)
      IF (SUM98-3.141592) 14,23,23
14    IF (SUM99-3.141592) 23,23,15
15    IF (SUM100-3.141592) 16,23,23
16    AV89=(SUM98+SUM99)/2.
      AV90=(SUM99+SUM100)/2.
      COMANS=(AV89+AV90)/2.
      IF (COMANS-3.1415920) 21,19,19
19    IF (COMANS-3.1415930) 20,21,21
20    WRITE(*,26)
      GO TO 22
21    WRITE(*,27) COMANS
22    STOP
23    WRITE(*,25)
      GO TO 22
25    FORMAT('ERROR IN MAGNITUDE OF SUM')
26    FORMAT('PROBLEM SOLVED')
27    FORMAT('PROBLEM UNSOLVED', F14.6)
28    FORMAT(I3, F14.6)
      END
```

- Golden hammer (золотой молоток) – одно решение

Code duplication (copy paste)

```
public void SetAudiType(Car car)
{
    car.Type = CarType.Audi;
}
public void SetBMwType(Car car)
{
    car.Type = CarType.Audi;
}
```

Magic numbers

if (database (ID) == 5)

константы, используемые в коде, но которые не несут никакого смысла без соответствующего комментария.

```
public void setPassword(String password) {
     // don't do this
     if (password.length() > 7) {
          throw new InvalidArgumentException("password");
     }
}
```

```
public static final int MAX_PASSWORD_SIZE = 7;

public void setPassword(String password) {
     if (password.length() > MAX_PASSWORD_SIZE) {
          throw new InvalidArgumentException("password");
     }
}
```

- Hard Code (жесткий код) - фиксация в коде различных данных об окружении.

Конфиг, база, константа

```
float areaOfCircle(int radius)
{
    float area = 0;
    area = 3.14*radius*radius;
    return area;
}
```

```
int main()
{
    const char *filename = "C:\\myfile.txt";

    printf("Filename is: %s\n", filename);
}
```

- Soft code

много абстракции

- Academicals complexity

(заумность решения)

- Boad anchor

(лодочный якорь)

сохранение неиспользуемых частей системы, которые остались после оптимизации или рефакторинга.

- Reinvention the wheal

(изобретение колеса)

Reinvention a square wheal

(пример: своя сортировка)

- Smell code

Старый код, недокументир.

- Blind faith (слепая вера)

(строка вместо числа, проверки внешних параметров) : недостаточная проверка корректности входных данных, отсутствие тестирования при разработки кода и исправлении ошибок.

- Бездумное комментирование

- God Object (божественный объект) (один класс)

Рефакторинг

- (англ. refactoring) или реорганизация кода — процесс изменения внутренней структуры программы, не затрагивающий её внешнего поведения и имеющий целью облегчить понимание её работы

Не оптимизация производительности

Не инженеринг

Причины применения

- \+ новую функцию, которая не добавляется в принятое архитектурное решение;
- есть ошибка, а причины не ясны;
- сложная логика программы

Признаки

- дублирование кода;
- длинный метод;
- большой класс;
- длинный список параметров;
- зависимые функции;
- избыточные временные переменные;
- не сгруппированные данные.

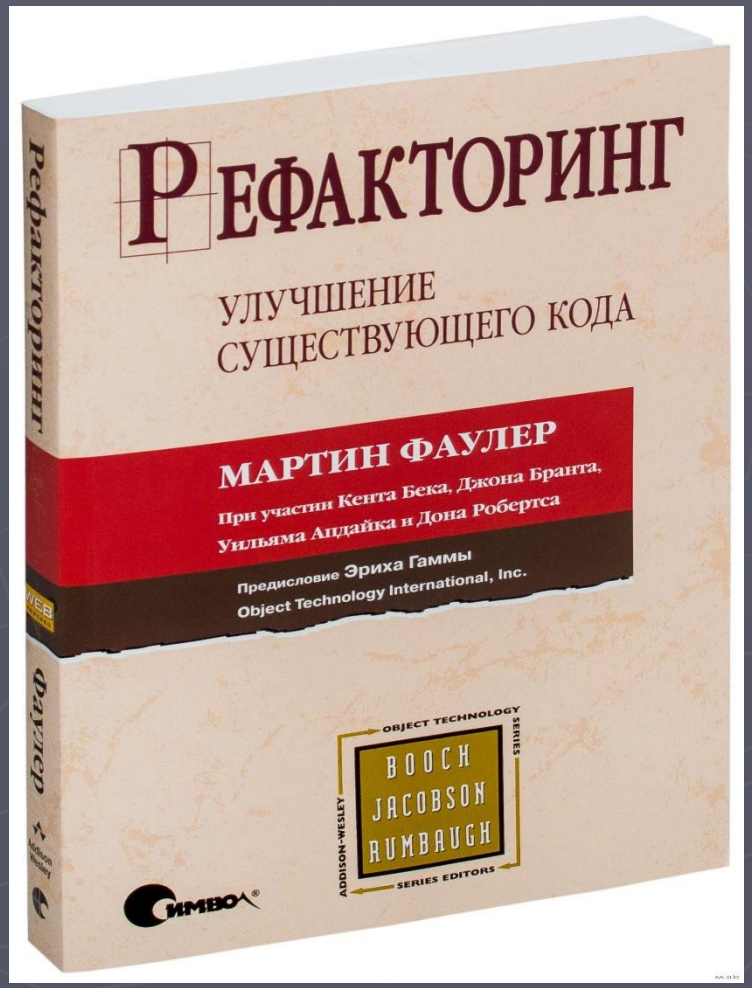
Методы рефакторинга

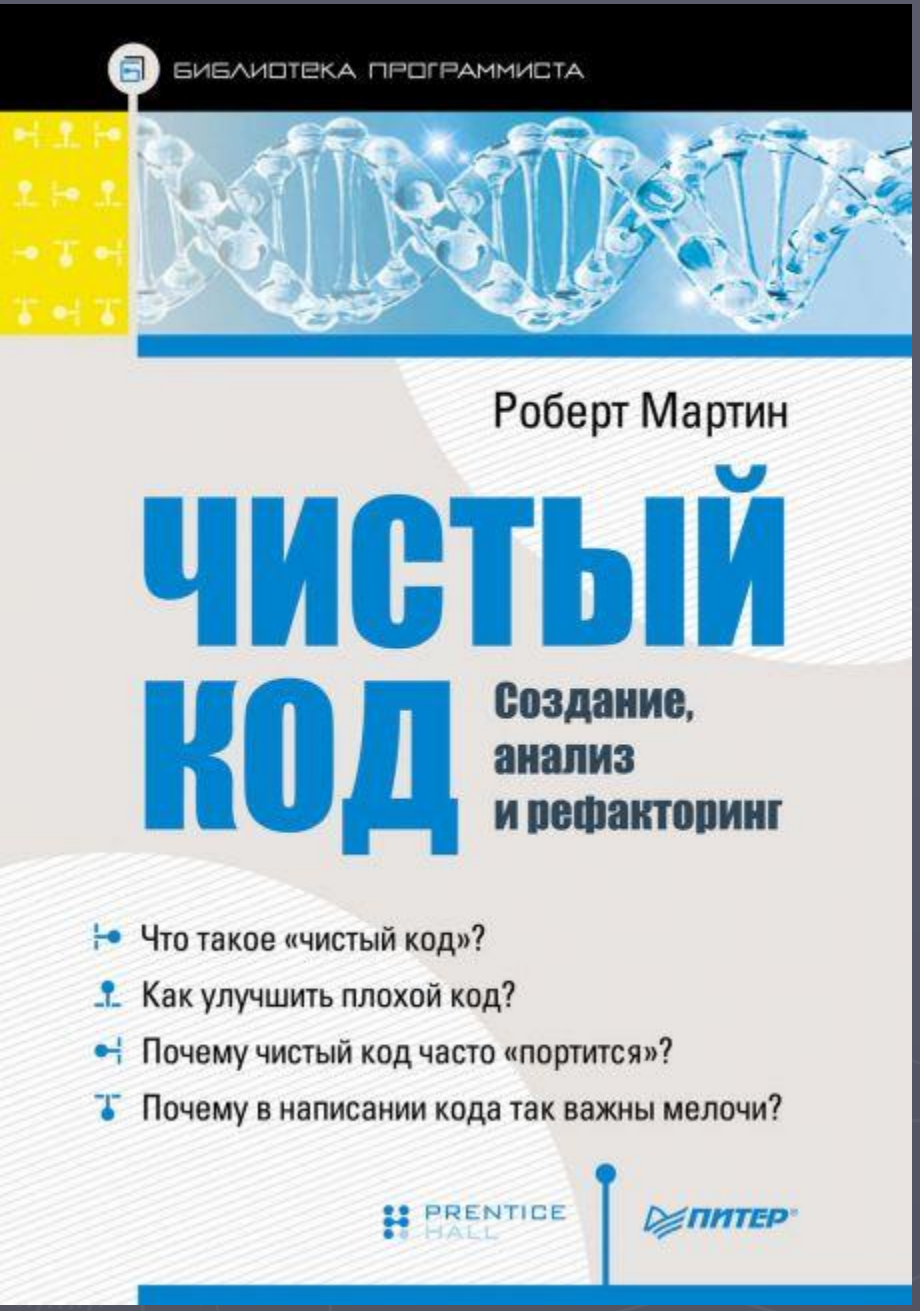
- Изменение сигнатуры метода
- Инкапсуляция поля
- Выделение класса
- Выделение интерфейса
- Выделение метода
- Встраивание (Inline)
- Введение фабрики

- Введение параметра
- Подъём метода
- Спуск метода
- Перемещение метода
- Замена условного оператора полиморфизмом
- Замена наследования делегированием
- И т .д.

- Фаулер М., Бек К., Брант Д., Робертс Д., Апдайк У.

Рефакторинг: улучшение существующего кода




БИБЛИОТЕКА ПРОГРАММИСТА
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?
PRENTICE HALL
ПИТЕР


ЧИСТЫЙ КОД
СОЗДАНИЕ, АНАЛИЗ И РЕФАКТОРИНГ
РОБЕРТ МАРТИН