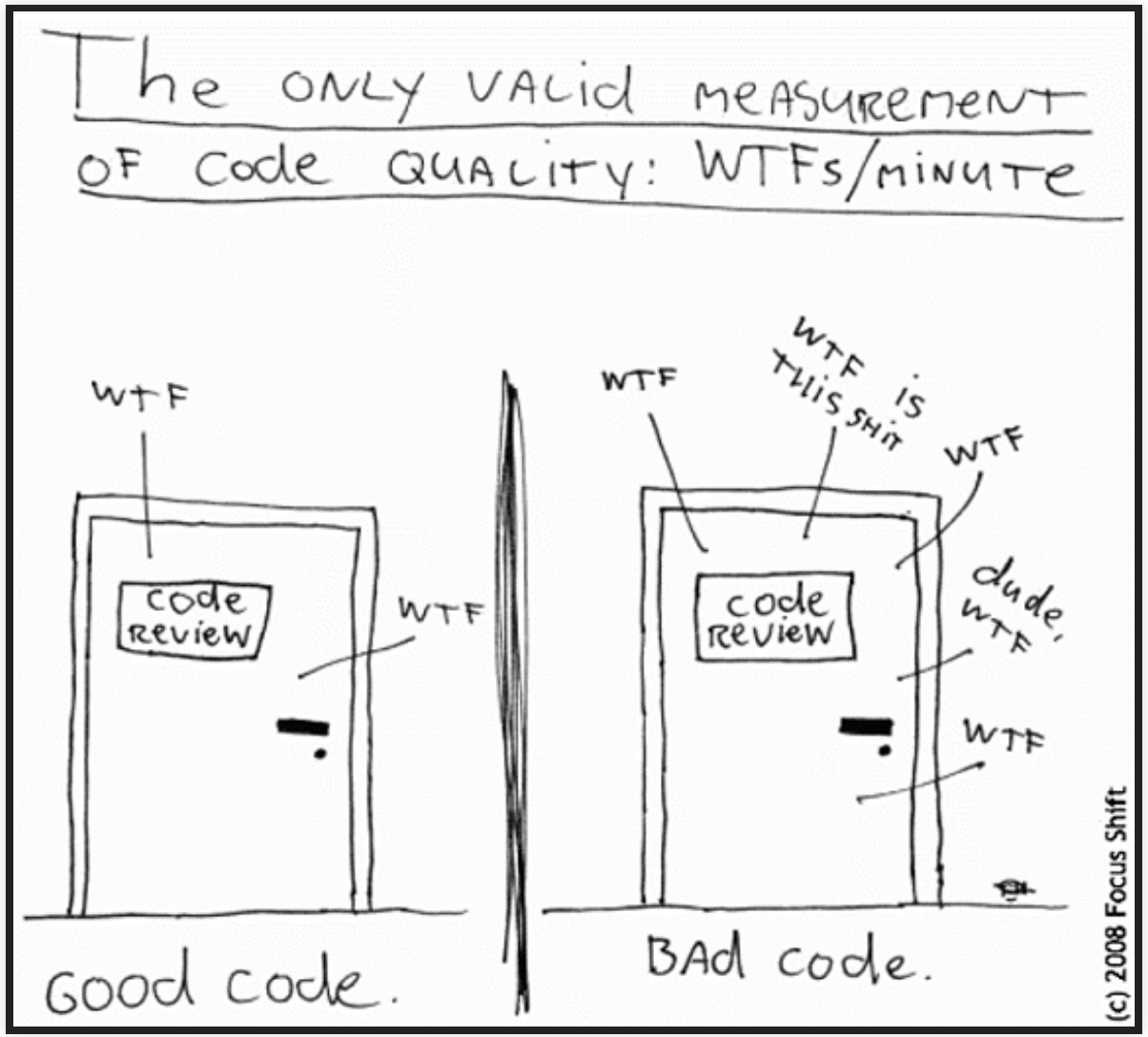
КАЧЕСТВЕН (CLEAN) CODE

13.11.2018

КАК МЕРИМ КАЧЕСТВОТО НА КОДА?



ПРИНЦИПИ НА ЧИСТИЯ КОД - СПАЗВАЙ ООП ПРИНЦИПИТЕ

- Енкапсулация
  - Свеждай публичните части до минимум
- Наследяване
  - Не допускай code duplication

ПРИНЦИПИ НА ЧИСТИЯ КОД - СПАЗВАЙ ООП ПРИНЦИПИТЕ

- Полиморфизъм
  - Ползвай полиморфизъм винаги, когато е възможно
  - Използвай интерфейс за декларация, конкретна имплементация за инициализация
- Абстракция

ПРИНЦИПИ НА ЧИСТИЯ КОД - СТРЕМИ СЕ КЪМ ХУБАВ ОО ДИЗАЙН

- Използвай само по изключение пакета по подразбиране
- Един клас трябва да прави едно нещо
- Ако имаш "and" в името, вероятно не е така
- Ако имаш "Helper", "Manager", "Utility" в името, *може би* има по-добър дизайн
- Един метод трябва да прави едно нещо
- Методът трябва да е кратък: < 20 реда код

ПРИНЦИПИ НА ЧИСТИЯ КОД - СТРЕМИ СЕ КЪМ ХУБАВ ОО ДИЗАЙН

- Ако имаш "and" в името на метод, може би той прави повечко неща: раздели го на няколко
- Ако имаш много параметри на метод:
  - може би не му е мястото в този клас (а там, откъдето се взимат стойностите за тези параметри)
  - или трябва да направиш data object, който да групира семантично свързаните от тях
- Не злоупотребявай със static

ПРИНЦИПИ НА ЧИСТИЯ КОД

- Форматирай си кода
- Нямаш никакво оправдание да не го правиш (IDE shortcut?)
- Вместо "магически числа" в кода, ползвай подходящо именувани константи

ПРИНЦИПИ НА ЧИСТИЯ КОД - СМИСЛЕНИ ИМЕНА

- Имената на пакети, класове, интерфейси, член-променливи, методи, локални променливи трябва да са говорящи и да спазват установените конвенции

ПРИНЦИПИ НА ЧИСТИЯ КОД - СМИСЛЕНИ ИМЕНА

- пакети
  - само малки букви, със смислена йерархия
  - `bg.sofia.uni.fmi.mjt`
- класове, абстрактни класове, интерфейси, enums
  - съществителни, започващи с главна буква (upper camel case)
  - `Student`, `GameBoard`

ПРИНЦИПИ НА ЧИСТИЯ КОД - СМИСЛЕНИ ИМЕНА

- методи
  - глаголи, започващи с малка буква (camel case)
  - `reverseString()`, `calculateSalary()`
- константи
  - all-caps, с подчертавки между думите
  - `MAX_NAME_LENGTH`

ПРИНЦИПИ НА ЧИСТИЯ КОД

```
// Bad
if (x % 2 == 0)
    return x / 2;
```

```
// Good
if (x % 2 == 0) {
    return x / 2;
}
```

- Винаги ограждай в блок телата на if-else и цикли, дори да са с по един statement

ПРИНЦИПИ НА ЧИСТИЯ КОД

```
// Bad
if (x % 2 == 0) {
    return true;
} else {
    return false;
}
```

```
// Good
return x % 2 == 0;
```

- Изразявай се кратко. Малко код == малко бъгове

ПРИНЦИПИ НА ЧИСТИЯ КОД

- Слагай коментари, без да прекаляваш
- Хубавият код е self-explanatory, и все пак, на места си трябва коментар
- Разделяй нормалната логика от exception логиката
- Ползвай изключения вместо error codes и печатане на съобщения в конзолата
- Не suppress-вай / swallow-вай exceptions
- Никога не оставяй празен catch, или catch само `e.printStackTrace()`

ПРИНЦИПИ НА ЧИСТИЯ КОД

- Разделяй I/O логиката от бизнес логиката

JAVA КОД КОНВЕНЦИИ

- Запознай се с цялостна Java код конвенция и се придържай към нея.
- Две от най-популярните:
- Oracle Code Conventions for the Java Programming Language
- Google Java Style Guide

ИНСТРУМЕНТИ ЗА СТАТИЧЕН КОД АНАЛИЗ

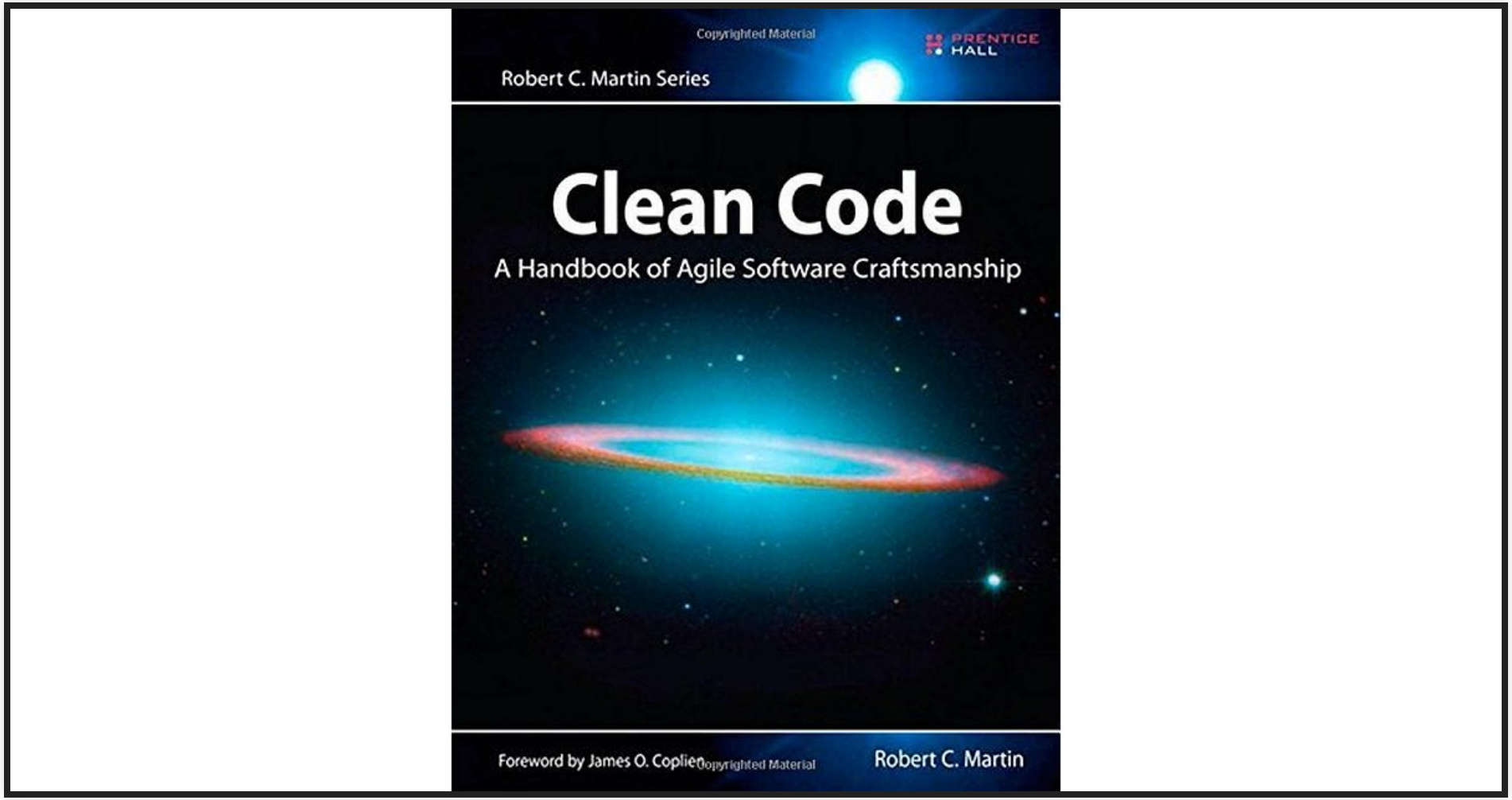
- Има инструменти за статичен код анализ, които
  - автоматизират придържането към код конвенции
  - намират и бъгове

ИНСТРУМЕНТИ ЗА СТАТИЧЕН КОД АНАЛИЗ

Някои от най-популярните open-source инструменти:

- checkstyle
- PMD
- FindBugs

БИБЛИЯТА


Copyrighted Material
PRENTICE HALL
Robert C. Martin Series
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Copyrighted Material
Robert C. Martin

PLANETGEEK

Clean Code Cheat Sheet

ВЪПРОСИ