



UNIVERSITATEA  
DIN BUCUREȘTI

# Metode de dezvoltare software

---

Inspectia codului

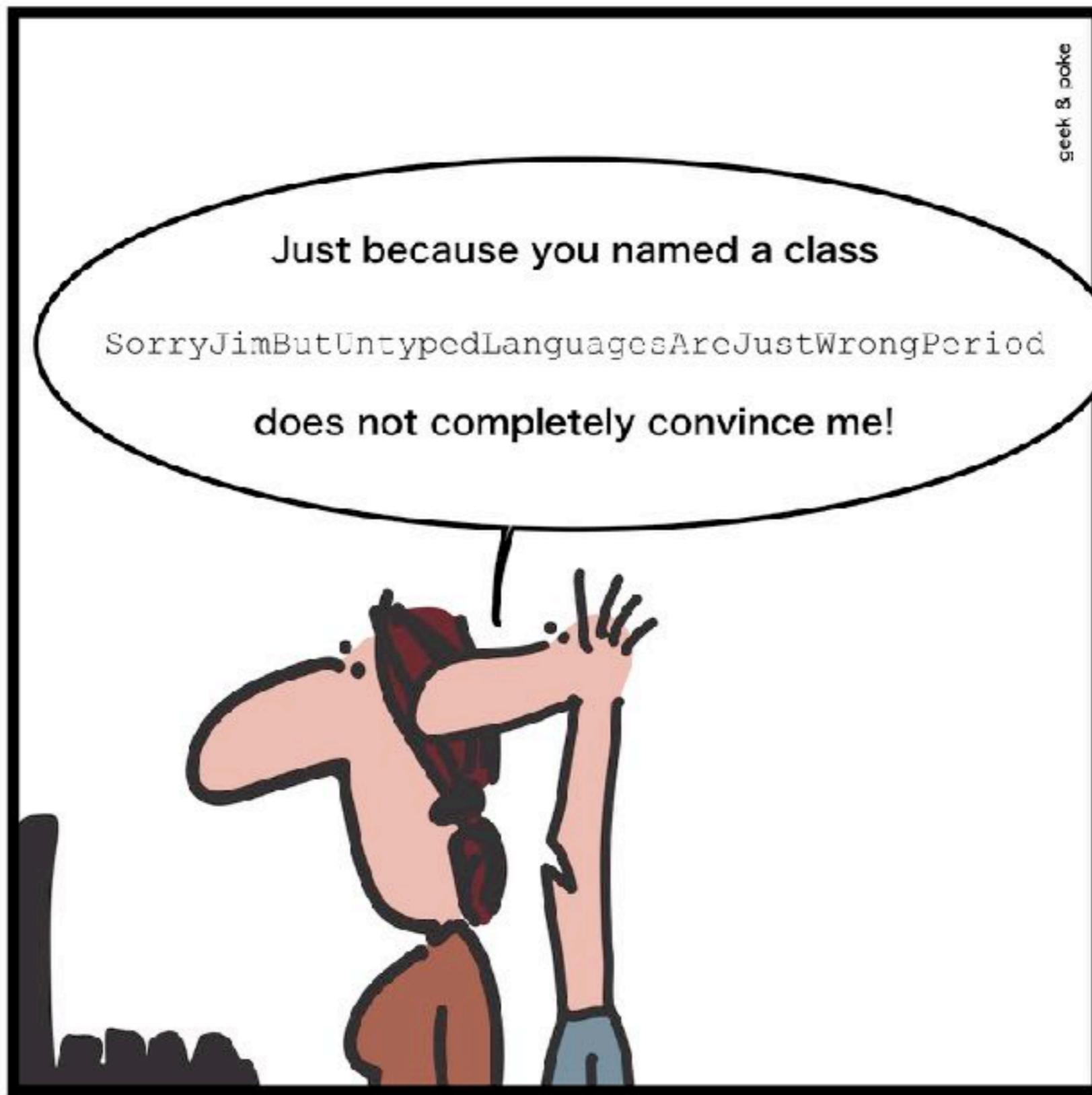
07.05.2018

Alin Ștefănescu

# **Inspectia codului**



# Code review... în practică



# Inspecțiile codului (code reviews)

- o metodă importantă pentru asigurarea calității este citirea codului cu scopul de a detecta erori
- În varianta extinsă, echipa de inspecție/recenzie a codului are patru membri:
  - Moderatorul – un programator competent
  - Programatorul – cel care a scris codul inspectat
  - Designer-ul, dacă este o persoană diferită de programator
  - Un specialist în testare

# Inspecțiile codului... în practică



# Inspeçțiile codului (2)

## ■ activități:

- programatorul citește, instrucțiune cu instrucțiune, logica programului
  - ceilalți participanți pun întrebări
- 
- programatorul însuși poate descoperi cele mai multe erori
  - se caută în program cele mai întâlnite tipuri de erori (folosind de exemplu o listă de erori comune)

# Inspeçțiile codului (3)

- inspectorii detectează erorile, programatorul trebuie să le corecteze
- dacă sunt descoperite multe erori sau unele erori necesită corecții substanțiale, moderatorul stabilește o nouă inspecție după corectarea lor
- erorile determinate pot conduce la modificarea design-ului
- de obicei, durata unei sesiuni de inspecție este de 90-120 de minute, cu o rată de 150 de instrucțiuni pe oră
- dacă programul este mai mare, se stabilesc mai multe sesiuni de inspecție, organizate pe module

# Inspeçțiile codului (4)

- programatorul nu trebuie să fie defensiv, ci constructiv
  - nu trebuie să vadă inspecția ca pe un atac personal, ci ca pe o modalitate de a crește calitatea muncii sale
  - rezultatele inspecției sunt confidențiale
- programatorul primește reacții privind stilul de programare, tehniciile și algoritmii folosiți
- ceilalți participanți beneficiază de pe urma expunerii la un alt stil de programare
- identificarea timpurie a secțiunilor celor mai predispuse la erori ajută la sporirea atenției acordate acestora în faza de testare

# Inspecțiile codului... varianta extinsă



# Observații

- există multe aspecte care pot fi luate în considerare:
  - formatul codului (d.ex. spații, taburi, locul parantezelor)
  - stilul programării (d.ex. variabilele sunt declarate la începutul metodei sau aproape de locul unde sunt folosite)
  - ce fel de nume sunt folosite (pentru clase, variabile, funcții)
  - acoperirea cu teste a codului inspectat

V. <https://blog.jetbrains.com/upsource/2015/07/23/what-to-look-for-in-a-code-review>

# Comparații cu alte metode

- inspectia codului completează procesul de testare și analiză statică a codului
- există paralele cu "pair programming", cu diferența principală că inspectia este realizată a posteriori și eventual asincron
- "code review" e o formă de "pair debugging"

# Tool-uri

- există tooluri care ajută procesul de inspecție a codului:

[https://en.wikipedia.org/wiki/List\\_of\\_tools\\_for\\_code\\_review](https://en.wikipedia.org/wiki/List_of_tools_for_code_review)

- in GitHub, inspectia este realizata cu un “pull request”
- de asemenea, există tooluri care ajută cu aspectele tehnice ale inspecției:
  - analiză statică a codului (code review automatizat)
  - verificarea sau formatarea urmărind anumite stiluri de programare

# Inspeția codului la diverse companii

- foarte multe companii folosesc intens inspecția codului
- la Google **nicio liniie de cod** nu este pusă în producție dacă nu a trecut printr-o inspecție. Vezi:
  - <http://goodmath.scientopia.org/2011/07/06/things-everyone-should-do-code-review/>
  - tooluri dezvoltate intern: Mondrian și mai nou, Critique. V. variante disponibile extern: Rietveld și Gerrit
- la Facebook, de asemenea e foarte important acest aspect:
  - tool dezvoltat intern: Phabricator
  - <https://phacility.com/phabricator/>

# Beneficii

- descoperirea de bug-uri
  - (citat din cartea "Code Complete" de McConnell): "... *software testing alone has limited effectiveness – the average defect detection rate is only 25 percent for unit testing, 35 percent for function testing, and 45 percent for integration testing. In contrast, the average effectiveness of design and code inspections are 55 and 60 percent.*"
- cod scris mai bine de la început, deoarece autorul știe că va fi citit și de altcineva
- transfer de cunoștințe
- se dorește obținerea corectitudinii programului
- soluții mai bune la o anumită problemă

# Posibile dezavantaje

- crește timpul investit în dezvoltarea de cod (cu speranța unui cod mai bun)
- cei care inspectează codul doar vânează greșeli minore sau doresc să-și impună punctul de vedere sau soluția la care se gândesc ei
- impune un "stres" suplimentar asupra dezvoltatorilor care preferă să lucreze singuri la anumite lucruri și au un anumit tip de introvertire (deși code reviews pot fi făcute asincron și offline)
- uneori, e suficient ca o persoană să-și facă singură inspecția codului și să găsească probleme
- v. și <https://dzone.com/articles/dont-waste-time-code-reviews>

# Exemplu - smelly code

## ■ de la <http://web.mit.edu/6.005/www/fa15/classes/04-code-review/>

```
public static int dayOfYear(int month, int dayOfMonth, int year)
{
    if (month == 2) {
        dayOfMonth += 31;
    } else if (month == 3) {
        dayOfMonth += 59;
    } else if (month == 4) {
        dayOfMonth += 90;
    } else if (month == 5) {
        dayOfMonth += 31 + 28 + 31 + 30;
    } else if (month == 6) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31;
    } else if (month == 7) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30;
    } else if (month == 8) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30 + 31;
    } else if (month == 9) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31;
    } else if (month == 10) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30;
    } else if (month == 11) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31;
    } else if (month == 12) {
        dayOfMonth += 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 31;
    }
    return dayOfMonth;
}
```

**rezolvați exercițiile online  
de la linkul de mai sus!**

# Scrierea de cod etic

- Inspectia codului poate preveni si detecta cod care "trișează" (nedetectat de teste)
- există din păcate și programatori (și manageri) care nu respectă anumite principii etice
- exemple recente de la Uber, Volkswagen sau Zenefits:  
<https://medium.freecodecamp.com/dark-genius-how-programmers-at-uber-volkswagen-and-zenefits-helped-their-employers-break-the-law-b7a7939c6591#.7r1eu6gne>