When does it smell too?

Lucerne, 14. February 2021
Mehdi Foudhaili

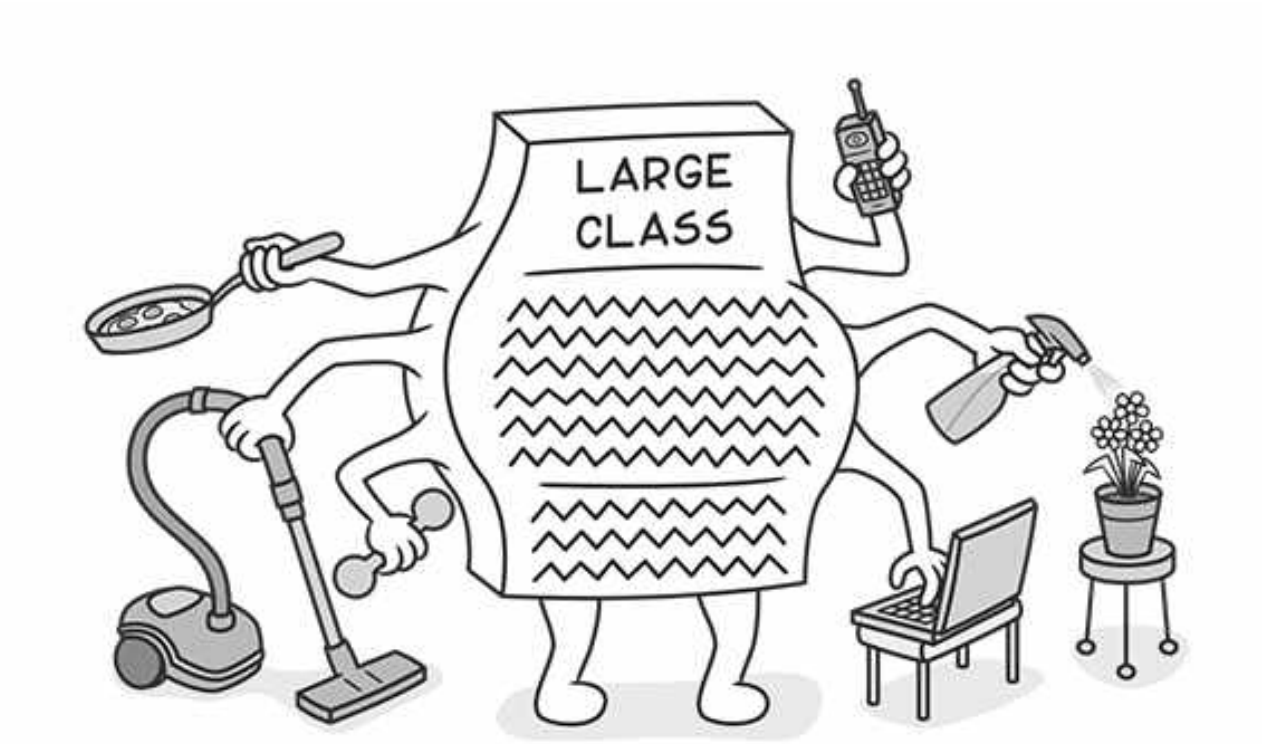
ALCOR Academy Training

Content

- What we have already learned
- Further Code Smells
- The Oddball Solution

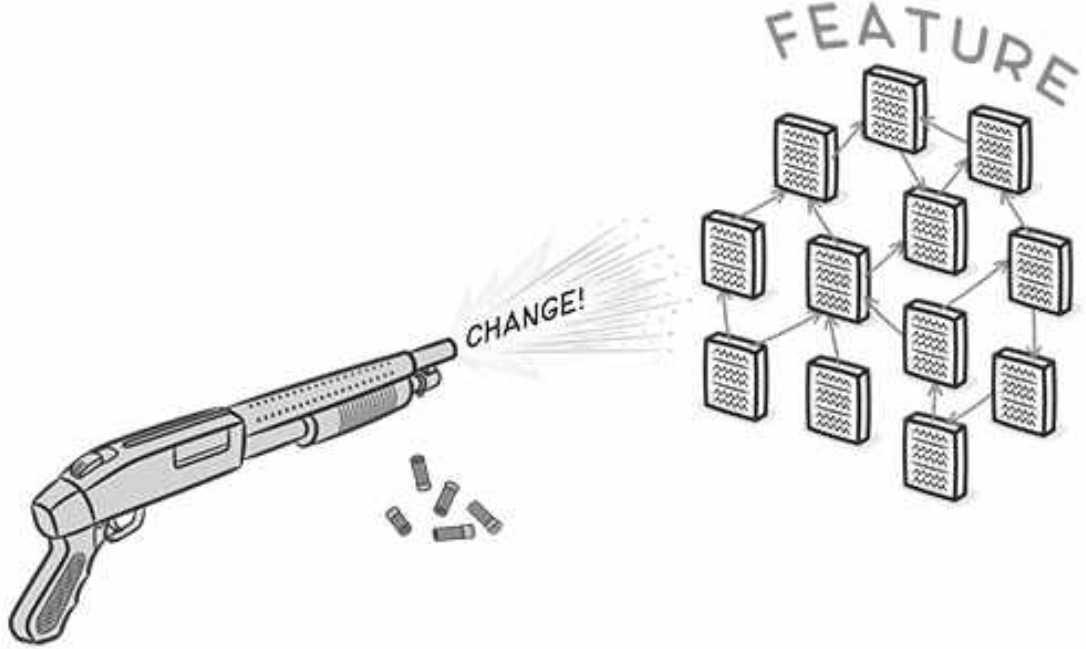
What do we have already learned?

- **Bloaters**
  - Long Method
  - Large Class
  - Primitive Obsession
  - Long Parameter List
  - Data Clumps



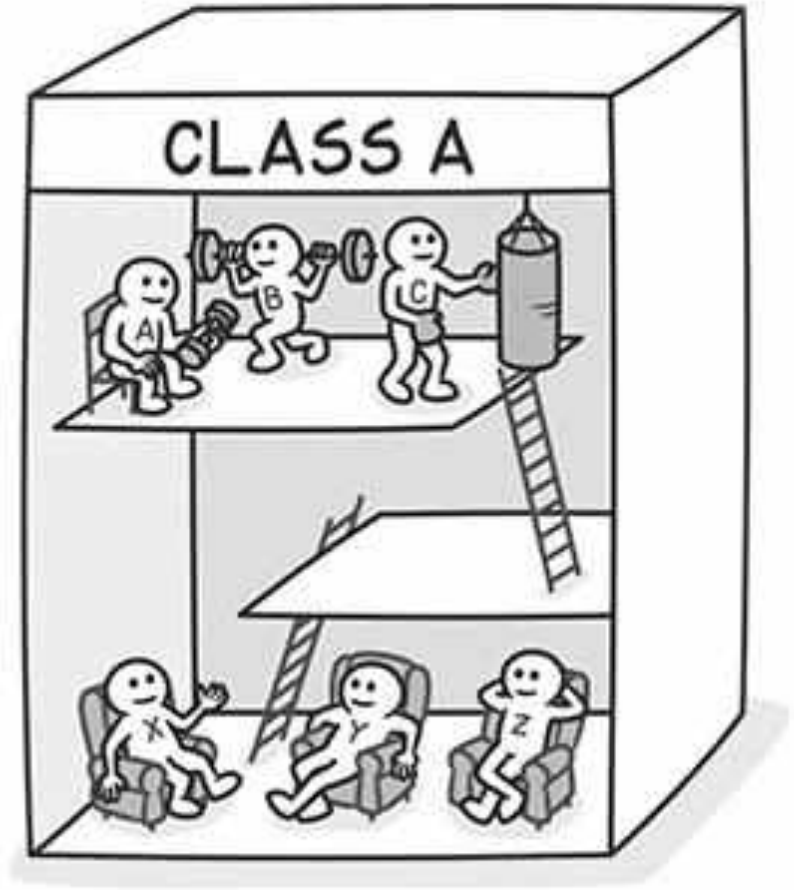
What do we have already learned?

- **Change Preventers**
  - Divergent Change
  - Shortgun Surgery
  - Parallel Inheritance Hierarch



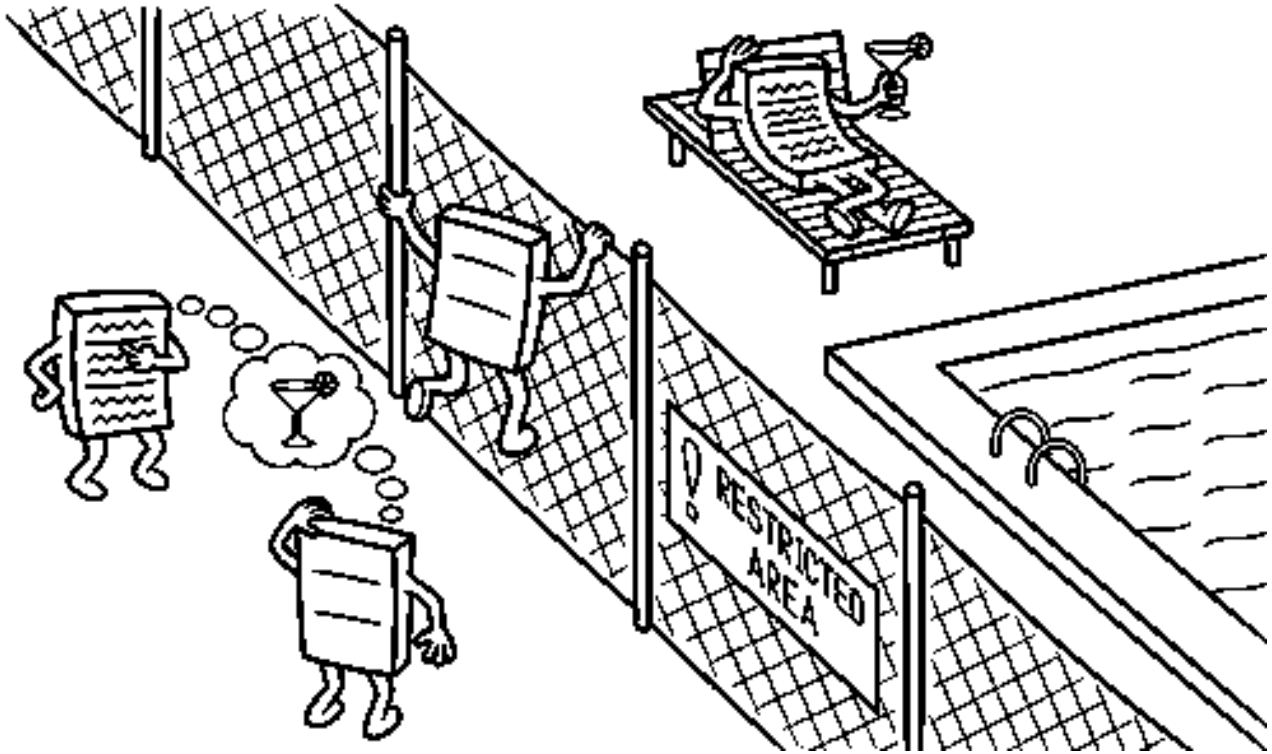
What do we have already learned?

- **OO Abusers**
  - Switch Statements
  - Temporary Field
  - Refused Bequest
  - Alternative Classes with Different Interfaces



What do we have already learned?

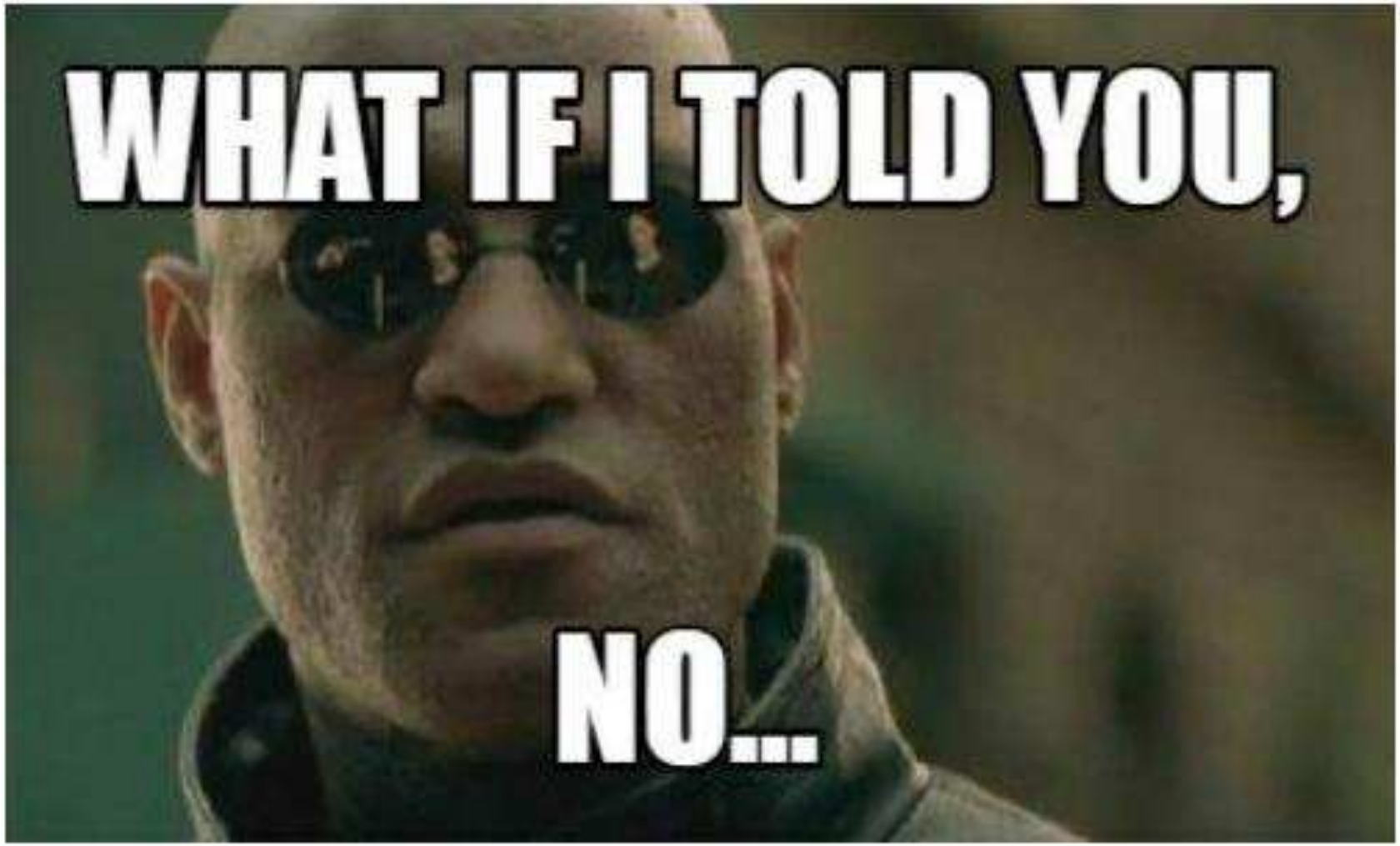
- **Couplers**
  - Feature Envy
  - Inappropriate Intimacy
  - Message Chains
  - Middle Man



What do we have already learned?

- **Dispensables**
  - Lazy Class
  - Data Class
  - Speculative Generality
  - Comments
  - Dead Code
  - Duplicated Code

Is that all???


WHAT IF I TOLD YOU,
NO...

Further Code Smells …

- **Bloaters**
  - Oddball Solution
  - Combinatorial Explosion
- **OO Abusers**
  - Class Depends on Subclass
  - Inappropriate Static
- **Couplers**
  - Artificail Coupling
  - Hidden Temporal Coupling
  - Hidden Dependencies
- **Even Tests could smell!!!**
  - https://deviq.com/antipatterns/code-smells#test-smells

Dispensables – Oddball Solution

- When a problem is solved in different way throughout a system.
- There should be only one way to solve the problem!

Dispensables – Oddball Solution

```
public class LoadUserProfileAction extends Action…
   public String process() throws Exception {
      …
      return process("ViewAction");
   }

public class UploadAction extends Action…
   public String process() throws Exception {
      …
      return process("ViewAction");
   }

public class ShowLoginAction extends Action…
   public String process() throws Exception {
      …
      return viewAction();
   }
```

Dispensables – Oddball Solution

- The presence of this code smell usually indicates:
  - Ignorance of the system
  - Programmers with different programming style and no existing standards
  - **Similar Classes with Different Interfaces!**
  - **Duplicated Code!**
- To get rid of this smell:
  - Just remove one of the solutions(generally the less used one)
  - Use the Adapter Pattern


THANK YOU FOR
YOUR ATTENTION
ANY QUESTIONS,
ASK GOOGLE

References & Links

- https://sourcemaking.com/refactoring/smells
- https://pragmaticways.com/31-code-smells-you-must-know/#6_Oddball_Solution
- https://pt.slideshare.net/nagarjay/code-smell-refactoring
- https://www.slideshare.net/MrinalBhattacaharya/code-smells-52370759
- https://blog.yellowoctopus.com.au/no-meme/
- https://medium.com/better-programming/what-is-it-that-makes-your-code-smell-f9c96ac93ba2#:~:text=An%20oddball%20solution%20is%20when,usually%20indicates%20subtly%20duplicated%20code.
- https://deviq.com/antipatterns/code-smells
- https://refactoringguru.cn/smells/