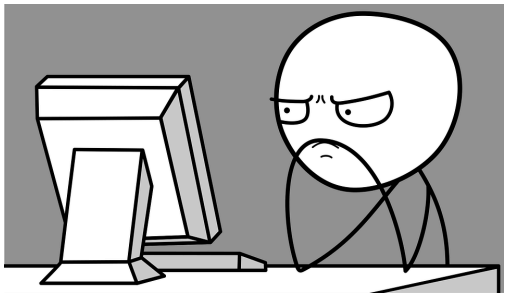
Anti-Patterns

Université Nice Sophia Antipolis
POLYTECH NICE-SOPHIA
Membre de UNIVERSITÉ CÔTE D'AZUR

Not Being Object Oriented



Classes without OO

- Static everywhere
- Breaking Law of Demeter : a.b().c().d()…

Abuse of utility class

- Only static methods

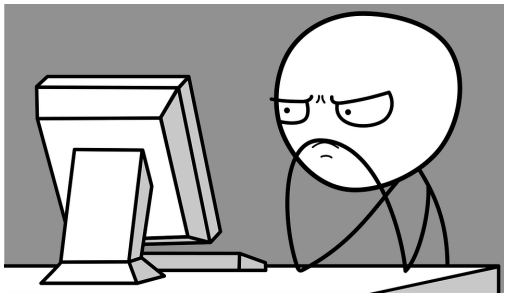
Misusing Inheritance

- Reusing code but no IS-A relationship is there

Not using polymorphism

- Using duplicate sets of 'if' (or switch/case) statements

Wrong OO Usage



Over Generalizing

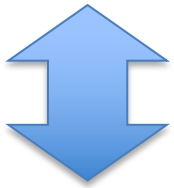
- Abstrating Classes to death

Object Orgy

- Attributes are public everywhere

God Class

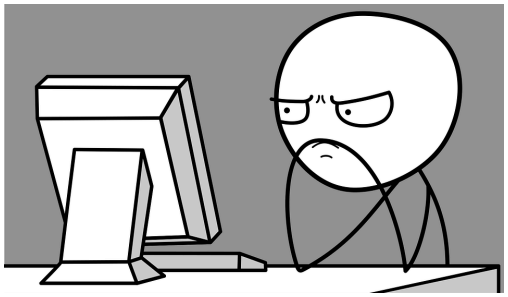
- *One Class to rule them all, and in the darkness bind them*



Poltegeist Class

- Useless classes with no real responsibility of their own, often used to just invoke methods in another class or add an unneeded layer of abstraction.

More General Anti-Patterns



Fear of Adding Classes

- 1 class will create unmanageable complexity ?!

Premature Optimization

- Optimizing before you have enough information to make educated conclusions about where and how to do the optimization.

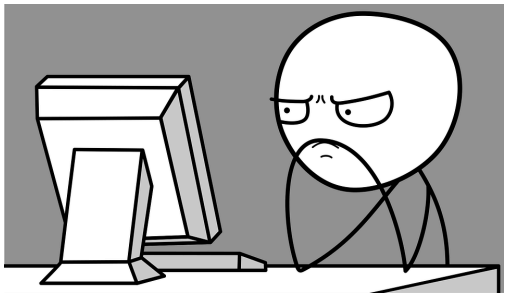
BikeShedding

- The law of futility

Analysis Paralysis

- Over-analyzing to the point that it prevents progress.

More General Anti-Patterns



Magic Numbers and Strings

- Using unnamed numbers or string literals instead of named constants in code

Cool, I'm going to use SONAR and drive the whole code quality with it

Management by Numbers

- Strict reliance on numbers for decision making

Sources

- http://wiki.c2.com/?ClassicOoAntiPatterns
- http://wiki.c2.com/?AntiPatternsCatalog

- http://sahandsaba.com/nine-anti-patterns-every-programmer-should-be-aware-of-with-examples.html