

Share  
this!

# CODE SMELLS THAT ARE FOUND THE MOST

12/12/2017 | IN AGILE WEB AND APP DEVELOPMENT | BY EKATERINA NOVOSELTSEVA

In [Apiumhub](#) we always focus on quality and best practices in [Software development](#). When we don't start working on a project we find code smells and this article is about it. Martin Fowler very well explained one day what is a code smell, it is a surface indication of a deeper problem in the software system. And the term was first coined by Kent Beck while helping Martin with the Refactoring book to read. Well, if you are interested in this topic, here you may find a list of other very useful [software development](#) and [software architecture](#) articles.

SEARCH ▾

[Apiumhub](#)

One of the nice things about code smells is that it's easy for inexperienced people to spot them, even if they don't know enough problem or to correct them. Many companies organize "code smells of the week" and ask developers to look for the smell and bring it to the team. Doing it one smell at a time is a good way of gradually teaching people on the team to be better programmers.

Share  
this!

## Common Code Smells

Developers are typically trained to look out for logical errors that have been accidentally introduced to their code. Such errors would be like null pointer exceptions that have not been handled to logical bugs that cause entire systems to crash. But what about the other issues that don't cause crashes, for example, the design issues that make the system hard to maintain, and increase the chance of bugs in the future, etc.? Code smells should be refactored in order to improve extendability, readability, and supportability.

Here are you have the most common code smells:

### Bloaters

Bloaters are code, methods and classes that have increased to such proportions that they are hard to work with. Usually these bloaters rather they accumulate over time as the program evolves. For example: Long Method, Large Class, Primitive Obsession, Long Parameter Lists, etc.

SEARCH

Apiumhub

## Object Orientation Abusers

All these smells are incomplete or incorrect application of object-oriented programming principles. For example, Switch Statement, Request, Alternative Classes with Different Interfaces

Share  
this!

### Change Preventers

These smells mean that if you need to change something in one place in your code, you have to make many changes in other places. It comes much more complicated and expensive as a result. For example: Divergent Change, Shotgun Surgery, Parallel Inheritance

### Dispensables

A dispensable is something pointless and unneeded whose absence would make the code cleaner, more efficient and easier to understand. Examples: Comments, Duplicate Code, Lazy Class, Data Class, Dead Code, Speculative Generality.

### Couplers

All the smells in this group contribute to excessive coupling between classes or show what happens if coupling is replaced by exception. Examples: Feature Envy, Inappropriate Intimacy, Message Chains, Middle Man, Incomplete Library Class.

Let's look at some of them in details, the ones that are found the most:



SEARCH

Apiumhub

## Long method

The majority of a programmer's time is spent reading code rather than writing code. Apart from the difficulty of having to keep a whilst reading through a long method, it is usually a sign that the method has too many responsibilities. Long methods make code it is not possible to view the whole method on your smartphone screen, consider breaking it up into several smaller methods, each

Share  
this!

## Duplicate Code

When developer fixes a bug, but same symptoms are faced again later on, this can be the result of code duplication, and a bug that is in the imperfect code but not in the duplicated versions. This poses an overhead in terms of maintenance. When developers are not fully aware of the occurrence they have come across. Take care of the repeated code blocks and extract them out into a single method.

## Inheritance method

If a class inherits from a base class but doesn't use any of the inherited fields or methods, developers should ask themselves if it is a good model. Signs of this code smell may be that the inherited methods go unused, or are overridden with empty method parts.

Inheritance should be used when a class wants to reuse the code in its superclass. If the classes diverge and the subclass no longer needs the inheritance, the hierarchy should be broken and delegation considered instead. And to keep some inheritance, remove the unused fields and methods and create a new layer that the objects can inherit from.

## Data Clumps

Where multiple method calls take the same set of parameters, it may be a sign that those parameters are related. To keep the code clean, it can be useful to combine them together in a class. This can help aid organisation of code.

SEARCH

Apiumhub

## Empty Shell

When a class exists just to delegate to another, a developer should ask themselves what its real purpose is. Sometimes this is the case where logic has been moved out of a class gradually, leaving an almost empty shell.

Share  
this!

## Primitive types

Primitive types give little in terms of domain context. Wrap them in a small class to represent the idea. Often this kind of class is added to the class.

## Divergent Code

This is when a class is commonly changed in different ways for different reasons and suffers many kinds of changes. So, ideally, you should separate common changes and classes.

## Shotgun Surgery

This is basically when you want to make a kind of change, you need to make a lot of little changes to a lot of different classes. The changes are all over the place, they are hard to find, and it's easy to miss an important change.

## Feature Envy

It is when a method does not leverage data or methods from the class it belongs to. Instead, it requires lots of data or methods



SEARCH ▾

Apiumhub

## Primitive Obsession

When you use multiple primitive data types to represent a concept such as using three integers to represent a date. Don't be afraid of tasks such as money classes that combine number and currency.

Share  
this!

## Lazy Class

A class that isn't doing enough to pay for itself, but remember that each class you create costs money to maintain and understand.

## Type Embedded in Name

Avoid placing types in method names; it's not only redundant, but it forces you to change the name if the type changes.

## Incommunicative Name

Does the name of the method succinctly describe what that method does? Could you read the method's name to another developer and understand what it does? If not, rename it or rewrite it. Pick a set of standard terminology and stick to it throughout your methods. For example, a button should probably have "Close".

## Dead Code

Delete code that isn't being used. Make it clean and simple.



SEARCH

Apiumhub

And if you are interested in best practices in software development, I highly recommend you to [subscribe to our monthly newsletter](#) development books, tips, and upcoming events.

Share  
this!

**If you liked this article about common code smells, you might like ...**

[Software development books to read in 2018](#)

[Why Kotlin language? Why did Google choose it?](#)

[S continuous integration with Fastlane & Jenkins](#)

[MVP pattern in iOS](#)

[Software architecture books to read this year](#)

[Scala Type bounds](#)

[F-bound over generic type of Scala](#)

[Charles Proxy in Android Emulator](#)

[Top software development blogs](#)



SEARCH ▾

[Apiumhub](#)

A Guide to TDD that will increase your productivity

Functional debt vs technical debt

BDD: UI Testing

Microservices vs monolithic architecture

Share  
this!

↳ RP in Object oriented design

↳ Almost infinit scalability

AGILE SOFTWARE DEVELOPMENT

PROGRAMMING

SOFTWARE DEVELOPER

TIPS



ADD COMMENT



SEARCH ▾

Apiumhub



---

Name

Email \*

Share  
this!

Website

POST COMMENT

---

## ABOUT APIUMHUB

---

Apiumhub is a Tech Hub that specialises in software architecture, mobile app & web development. Here we share with you industry tips & best practices.

## CATEGORIES

---

SEARCH

Apiumhub

Share  
this!

- > Events (29)
- > Jobs (3)
- > Offshoring and outsourcing (5)
- > Product Ownership (8)
- > Software architecture (23)
- > Technology industry trends (104)
- > Tools (4)
- > User Experience Design (52)


#### JOIN OUR NEWSLETTER

SUBSCRIBE

< PREV

SEARCH

Apiumhub



Apiumhub is a software development company based in Barcelona that transformed into a tech hub, mainly offering services of mobile app development, web development & software architecture.

Share  
this!

JBCNConf 2019: The Pursuit of Pragmatic  
Programming

06/20/2019

JBCNConf 2019 Highlights: Evolving a Clean  
Architecture

06/18/2019

mDevCamp Conference 2019: our experience

06/17/2019

Type email

**SUBSCRIBE**

