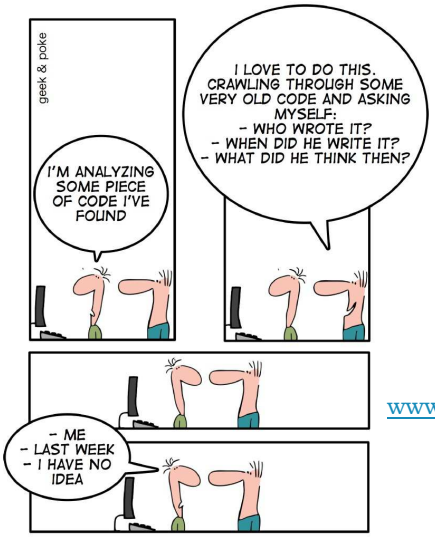
0x01

Coders love good code



Inspired by a Slashdot post:
http://ask.slashdot.org/article.pl?sid=08/01/18/1554257

Responsibility

www.cs.uoi.gr/~zarras/soft-devII.htm

Sources: M. Fowler Refactorings Catalog
W. Wake Refactoring Workbook

Feature Envy

Feature Envy

Symptoms

You see a method in a class that seems to be focused **on manipulating the data of another class rather than itself**. (You may notice this because of duplication—several clients do the same manipulation.)

Causes

This is very common among **clients** of current and former **data classes**, but you can see it for any class and its clients.

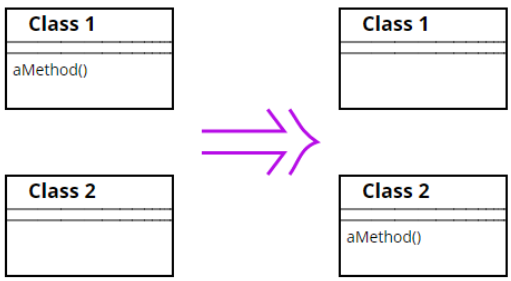
What to Do

Move Method *to put the actions on the correct class. (You may have to* ***Extract Method*** *first to isolate the misplaced part.)*

Move Method

A method is, or will be, using or used by more features of another class than the class on which it is defined.

Create a new method with a similar body in the class it uses most. Either turn the old method into a simple delegation, or remove it altogether.



Check the sub- and superclasses of the source class for other declarations of the method.

If there are any other declarations, you may not be able to make the move, unless the polymorphism can also be expressed on the target.

Divergent Change

Divergent Change

Symptoms

You find yourself **changing the same class** for **different reasons**.

Causes

The class picked up **more responsibilities** as it evolved, with no one noticing that two different types of decisions were involved.

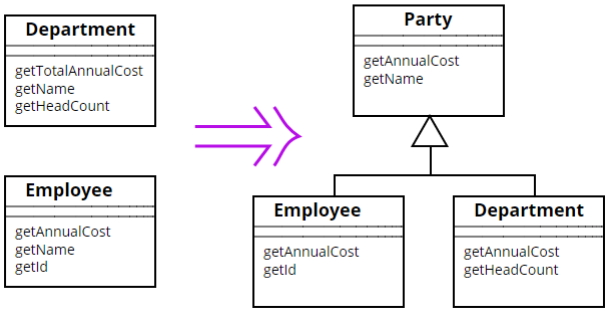
What to Do

1. ***Extract Class*** *to pull out separate classes for the separate responsibilities.*
2. If several classes are sharing the same type of decisions, you may be able to consolidate those new classes (e.g., by ***Extract Superclass*** *or* ***Extract Subclass***).

Extract Super Class

You have two classes with similar features.

Create a superclass and move the common features to the superclass.



Extract Super Class

Another form of duplicate code is two classes that do similar things in the same way or similar things in similar ways.

Objects provide a built-in mechanism to simplify this situation with inheritance. However, often we don't notice the commonalities until we have created some classes, in which case we need to create the inheritance structure later.

It's usually easier to move the common fields first.

If you have subclass methods that have different signatures but the same purpose, use Rename Method to get them to the same name and then use Pull Up Method.

If you have methods with the same signature but different bodies, declare the common signature as an abstract method on the superclass.

Shotgun Surgery

Shotgun Surgery

Symptoms

You want to make a "**simple**" change, but you have to **touch several classes**.

Causes

One responsibility has been **split** among **several classes**. There may be a missing class that would understand the whole responsibility.

What to Do

1. **Identify the class** that should **own** the **group of changes**. It may be an existing class, or you may need to *Extract Class to create a new one.*
2. Use *Move Field and Move Method to put the functionality onto the chosen class.*