Object Calisthenics Cheat Sheet

by museumoftechno (museumoftechno) via cheatography.com/19333/cs/2395/

Wrap primitives and strings in objects

An **Hour** object's type-hint in the parameters of a function tells you about the meaning of that parameter

You can add relevant methods to wrapper objects, cutting down on duplication - EG a *round()* function as part of a **Money** class

Don't abbreviate

Is a method name too long? Maybe it is doing too much

Are you having to type the method name a lot? Maybe you're missing an opportunity to de-duplicate?

Document code

Docblocks

No more than two instance variables per class

If a class is dealing with lots of instance variables, it's probably doing too much

If a class has lots of dependencies, maybe moving responsibilities from one dependency to another might help eliminate dependencies

One indentation level per method

Promotes method cohesiveness

Try to ensure each method does exactly one thing

Methods may well become better named as they become simpler and their responsibilities more focused

First class collections

Every collection is wrapped in its own class

Methods (EG filters) proper to a collection can be added to the wrapper class

Processing that needs to be carried out on every member of a collection can go in the wrapper class

Keep all entities small

Each method <= 5 lines

Each class <= 50 lines (~10 methods)

Each package <= 10 files

Methods, classes AND PACKAGES should be cohesive

Don't use getters/setters

Tell, don't ask

Don't use ELSE

Favour polymorphism over conditionals

Should make classes smaller, simpler, tighter

Also try Null Object pattern

One dot per line

If an object is juggling lots of other objects, it knows too much

If you're chaining together lots of calls to objects to return yet other objects (or grab them from object properties), then acting on those objects in turn... that's a code smell

Consider moving processing from current object into other objects

If you're digging deeply into another object and rearranging it, you're violating encapsulation

You can play with your toys, toys that you make, and toys that someone gives you. You don't play with your *toy's* toys

By **museumoftechno** (museumoftechno)

cheatography.com/museumoftechno/

Published 28th July, 2014.
Last updated 7th August, 2014.