

5. OBJECTS AND DATA STRUCTURES



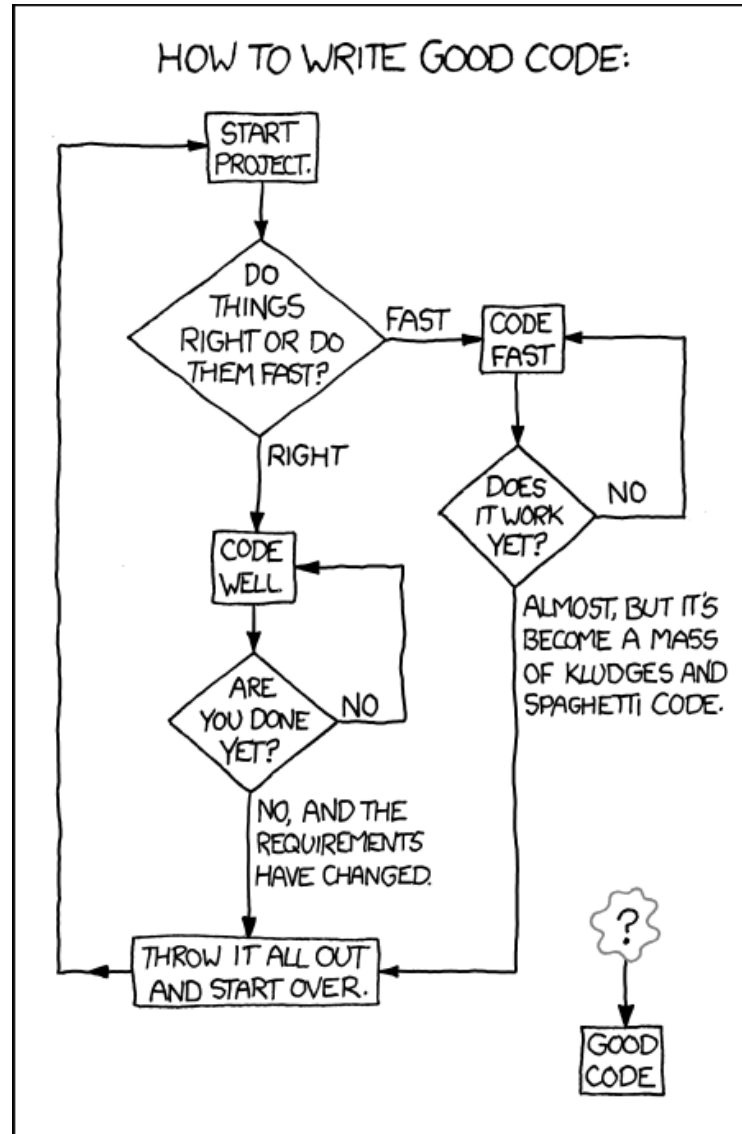
WRITING CLEAN CODE

By Pablo Restrepo

Based on the book
"Clean Code" by Uncle
Bob



Bad Code



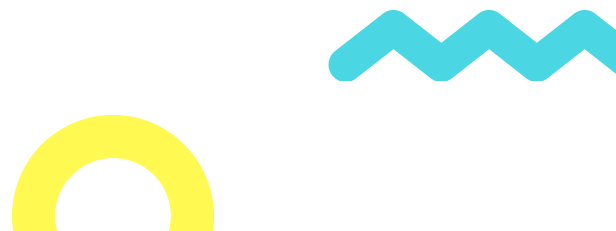


Bad Code

Have you ever be slowed down by messy code?.

have you ever waded through a mess so grave that it took weeks to do what should have taken hours?.

have you seen what should have been a one-line change made instead in hundreds of different modules?.





Bad Code

I bet that doesn't happen at
your organization!

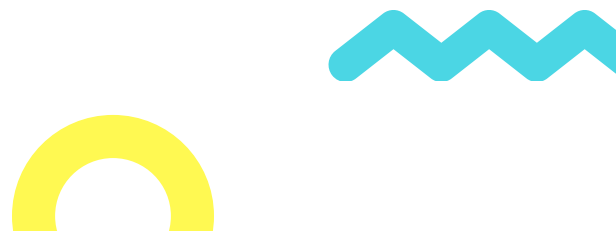




Bad Code

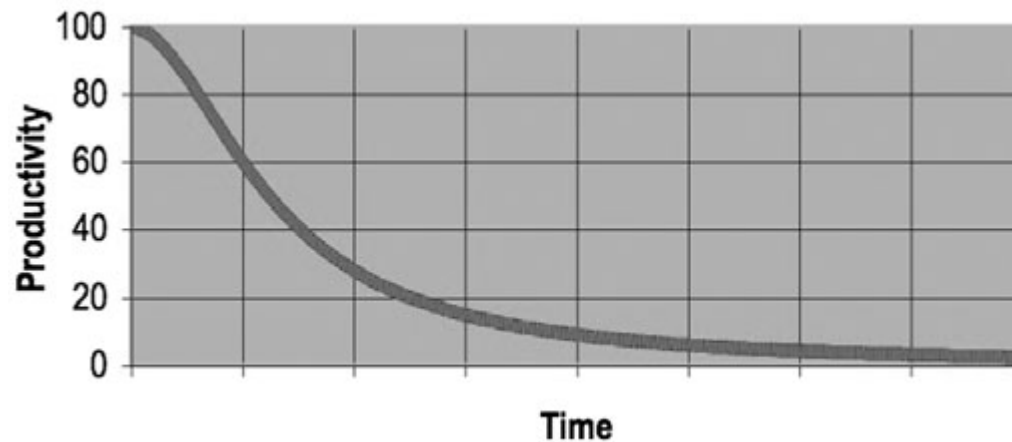
The responsibility for keeping the code clean is ours!. We are the ones to blame.

Managers rely on us to tell them how long projects should take in order to maintain the code clean.



The cost of owning a mess

We need to take the time to be fast






The cost of owning a mess

Leblanc's law: Later equals never






The boy scout rule

Leave it a little bit cleaner than you found
it






The art of Clean Code

Code is like a painting: you can recognize if it's good or not, but that doesn't mean you know how to paint a good picture.

Let's learn how to write good code.





2. MEANINGFUL NAMES




Use Intention revealing names

If a name requires a comment, then it does
not reveal its intent

```
int d; // elapsed time in days
```

```
int elapsedTimeInDays;  
int daysSinceCreation;  
int daysSinceModification;  
int fileAgeInDays;
```

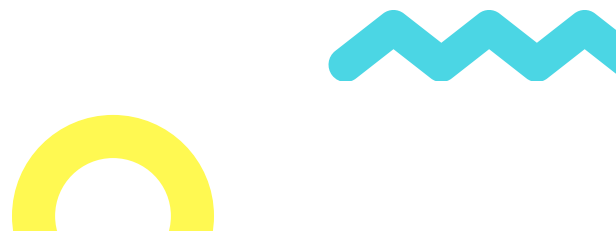




Use Intention revealing names

What is the purpose of this code?:


<https://github.com/prestrepoh/Clean-Code-Course/blob/master/1-meaningful%20names/1-use-intention-revealing-names/badnamedlist.java>





Use Intention revealing names

Why is it difficult to understand what it
does?

1. What kind of things are in theList?
 2. What is the importance of the zeroth
subscript of an item in theList?
 3. What is the significance of value 4?
 4. How would I use the list being returned?
- 



Use Intention revealing names

Why if the previous code was a function for a mine sweeper game?. Wouldn't it be clearer if wrotten like this:

<https://github.com/prestrepoh/Clean-Code-Course/blob/master/1-meaningful%20names/1-use-intention-revealing-names/minesweeper1.java>





Use Intention revealing names

What if we go further and write a simple class for cells instead of using an array of ints? Then we could use the intention-revealing function `isFlagged` to hide the magic numbers

<https://github.com/prestrepoh/Clean-Code-Course/blob/master/1-meaningful%20names/1-use-intention-revealing-names/minesweeper2.java>





Avoid Disinformation

hp is not a good name for a hypotenuse. It
can be disinformative.

Do not refer to a grouping of accounts as an
accountList unless it is in fact a ***List***

Beware of using names which vary in small
ways. We programmers love to use
autocomplete.

XYZControllerForEfficientHandlingOfStrings
XYZControllerForEfficientStorageOfStrings





Avoid Disinformation

Try not to use lowercase l or uppercase O as variable names in contexts where they can be confused:

```
int a = 1;  
if ( O == 1 )  
    a = 01;  
else  
    l = 01;
```





Make Meaningful Distinctions


Don't name variables only to satisfy the compiler
(zork vs thezork. Klass vs Class)

The names a1, a2 are not disinformative, but they are
noninformative

what's the difference between the
variables: moneyAmount and money?

whats the difference between the funcitons:

```
getActiveAccount();  
getActiveAccounts();  
getActiveAccountInfo();
```







Use Pronounceable Names

```
class DtaRcrd102 {  
    private Date genymdhms;  
    private Date modymdhms;  
    private final String pszqint = "102";  
    /* ... */  
};
```

vs

```
class Customer {  
    private Date generationTimestamp;  
    private Date modificationTimestamp;  
    private final String recordId = "102";  
    /* ... */  
};
```





Use Searchable Names

One might easily grep for
`MAX_CLASSES_PER_STUDENT`, but the number
7 could be more
troublesome

The name `e` is a poor choice for any variable
for which a programmer might
need to search

Single-letter names can ONLY be used as
local variables
inside short methods






Use Searchable Names

If a variable or constant might be seen or used in multiple places in a body of code, it is imperative to give it a search-friendly name. Once again compare

```
for (int j=0; j<34; j++) {  
    s += (t[j]*4)/5;  
}  
to  
int realDaysPerIdealDay = 4;  
const int WORK_DAYS_PER_WEEK = 5;  
int sum = 0;  
for (int j=0; j < NUMBER_OF_TASKS; j++) {  
int realTaskDays = taskEstimate[j] * realDaysPerIdealDay;  
int realTaskWeeks = (realdays / WORK_DAYS_PER_WEEK);  
sum += realTaskWeeks;  
}
```





Avoid Encodings

Just avoid them!






Avoid Mental Mapping

Readers shouldn't have to mentally translate your names into other names they already know

A loop counter should use `i`, `j` and `k` because they are traditional, but it would be even better if they had revealing names!

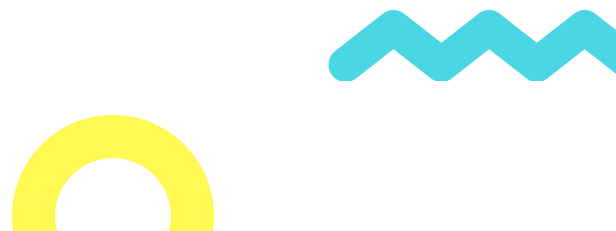
There can be no worse reason for using the name `c` than because `a` and `b` were already taken





Class Names

Classes and objects should have noun or noun phrase names like Customer, WikiPage, Account, and AddressParser. Avoid words like Manager, Processor, Data, or Info in the name of a class. A class name should not be a verb.





Method Names

Methods should have verb or verb phrase names like `postPayment`, `deletePage`, or `save`.

Mutators should be named with `get` and `set`

When constructors are overloaded, use static factory methods with names that describe the arguments.

```
Complex fulcrumPoint =  
Complex.FromRealNumber(23.0);
```

is generally better than

```
Complex fulcrumPoint = new Complex(23.0);
```





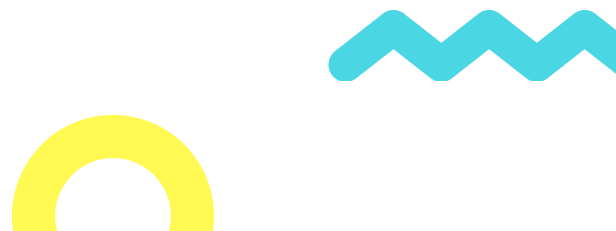
Don't Be Cute


What is ***HolyHandGrenade*** is supposed to do?

Sure,

it's cute, but maybe in this case
`DeleteItems` might be a better name.

Don't tell little culture-dependent jokes like
`eatMyShorts()` to mean `abort()`

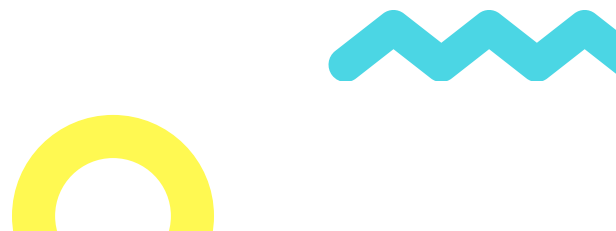





Pick One Word per Concept

Pick one word for one abstract concept and stick with it. For instance, it's confusing to have `fetch`, `retrieve`, and `get` as equivalent methods of different classes.

It's confusing to have a `controller` and a `manager` and a `driver` in the same code base.

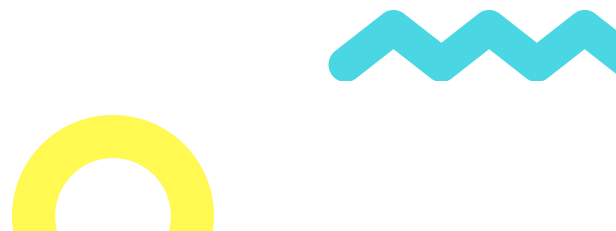




Don't Pun

If in one class "add" is used to concatenate Strings, it would be confusing to use it in another class to add an element to a list. Be consistent!

We want to use the popular paperback model whereby the author is responsible for making himself clear and not the academic model where it is the scholar's job to dig the meaning out of the paper.





Use Solution Domain Names

Remember that the people who read your code will be programmers. So go ahead and use computer science (CS) terms, algorithm names, pattern names, math terms, and so forth.

The name AccountVisitor means a great deal to a programmer who is familiar with the VISITOR pattern

