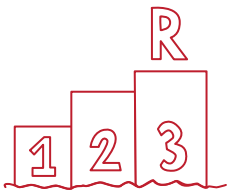
When to refactor

Rule of Three



1. When you're doing something for the first time, just get it done.
2. When you're doing something similar for the second time, cringe at having to repeat but do the same thing anyway.
3. When you're doing something for the third time, start refactoring.

When adding a feature

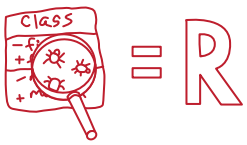
- Refactoring helps you understand other people's code. If you have to deal with someone else's dirty code, try to refactor it first. Clean code is much easier to grasp. You will improve it not only for yourself but also for those who use it after you.
- Refactoring makes it easier to add new features. It's much easier to make changes in clean code.

When fixing a bug

Bugs in code behave just like those in real life: they live in the darkest, dirtiest places in the code. Clean your code and the errors will practically discover themselves.

Managers appreciate proactive refactoring as it eliminates the need for special refactoring tasks later. Happy bosses make happy programmers!

During a code review



The code review may be the last chance to tidy up the code before it becomes available to the public.

It's best to perform such reviews in a pair with an author. This way you could fix simple problems quickly and gauge the time for fixing the more difficult ones.

Tired of reading?

No wonder, it takes 7 hours to read all of the text we have here.

Try our interactive course on refactoring. It offers a less tedious approach to learning new stuff.

Let's see...

READ NEXT

How to refactor