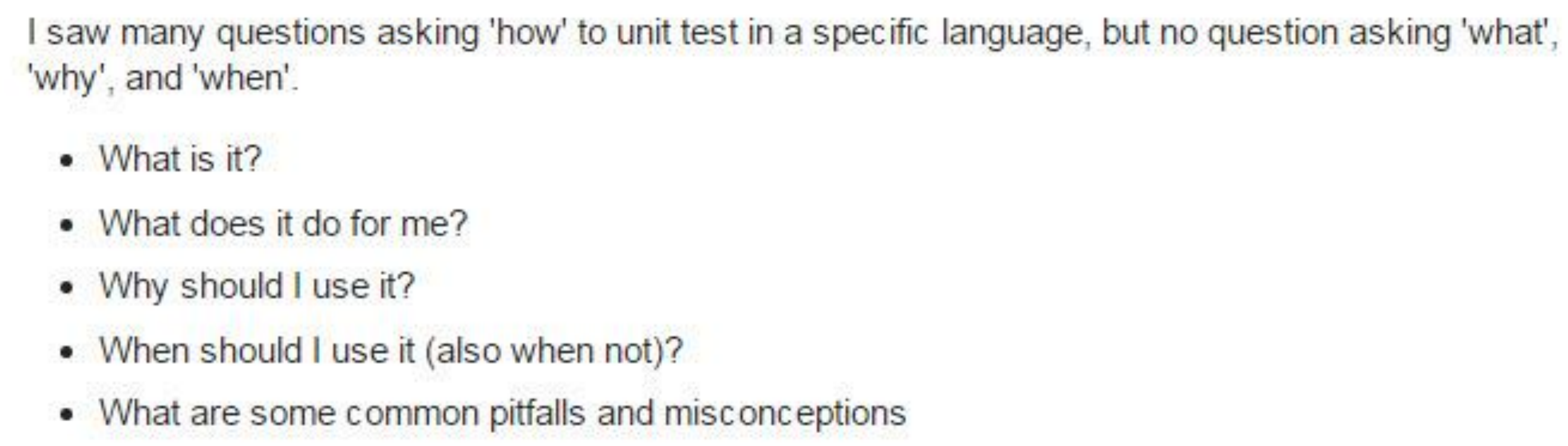
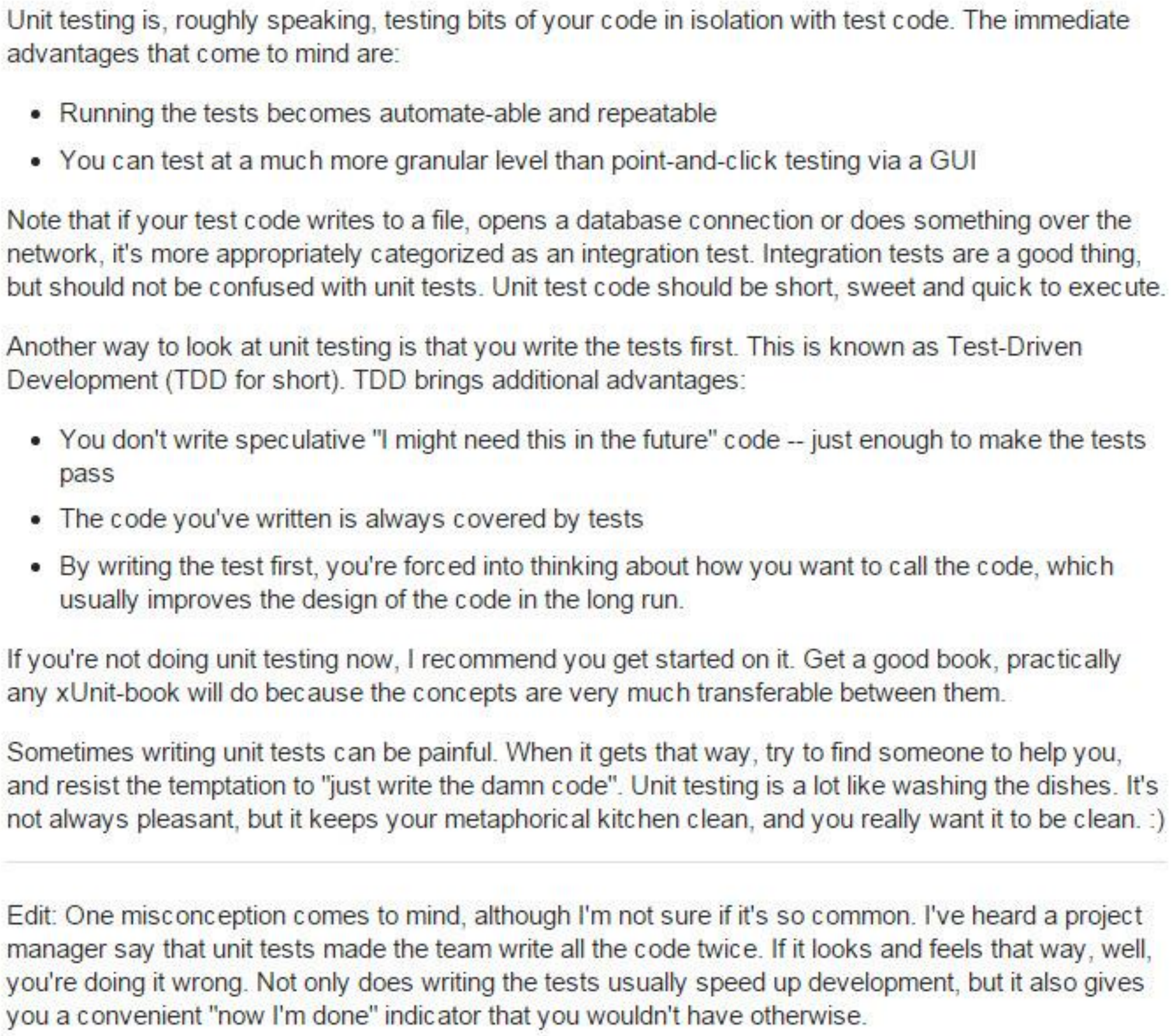
Why Do Unit Testing?



I saw many questions asking 'how' to unit test in a specific language, but no question asking 'what', 'why', and 'when'.

- What is it?
- What does it do for me?
- Why should I use it?
- When should I use it (also when not)?
- What are some common pitfalls and misconceptions

Answers, which are taken from Stack Overflow, *appear on the next page.*



Unit testing is, roughly speaking, testing bits of your code in isolation with test code. The immediate advantages that come to mind are:

- Running the tests becomes automate-able and repeatable
- You can test at a much more granular level than point-and-click testing via a GUI

Note that if your test code writes to a file, opens a database connection or does something over the network, it's more appropriately categorized as an integration test. Integration tests are a good thing, but should not be confused with unit tests. Unit test code should be short, sweet and quick to execute.

Another way to look at unit testing is that you write the tests first. This is known as Test-Driven Development (TDD for short). TDD brings additional advantages:

- You don't write speculative "I might need this in the future" code -- just enough to make the tests pass
- The code you've written is always covered by tests
- By writing the test first, you're forced into thinking about how you want to call the code, which usually improves the design of the code in the long run.

If you're not doing unit testing now, I recommend you get started on it. Get a good book, practically any xUnit-book will do because the concepts are very much transferable between them.

Sometimes writing unit tests can be painful. When it gets that way, try to find someone to help you, and resist the temptation to "just write the damn code". Unit testing is a lot like washing the dishes. It's not always pleasant, but it keeps your metaphorical kitchen clean, and you really want it to be clean. :)

Edit: One misconception comes to mind, although I'm not sure if it's so common. I've heard a project manager say that unit tests made the team write all the code twice. If it looks and feels that way, well, you're doing it wrong. Not only does writing the tests usually speed up development, but it also gives you a convenient "now I'm done" indicator that you wouldn't have otherwise.