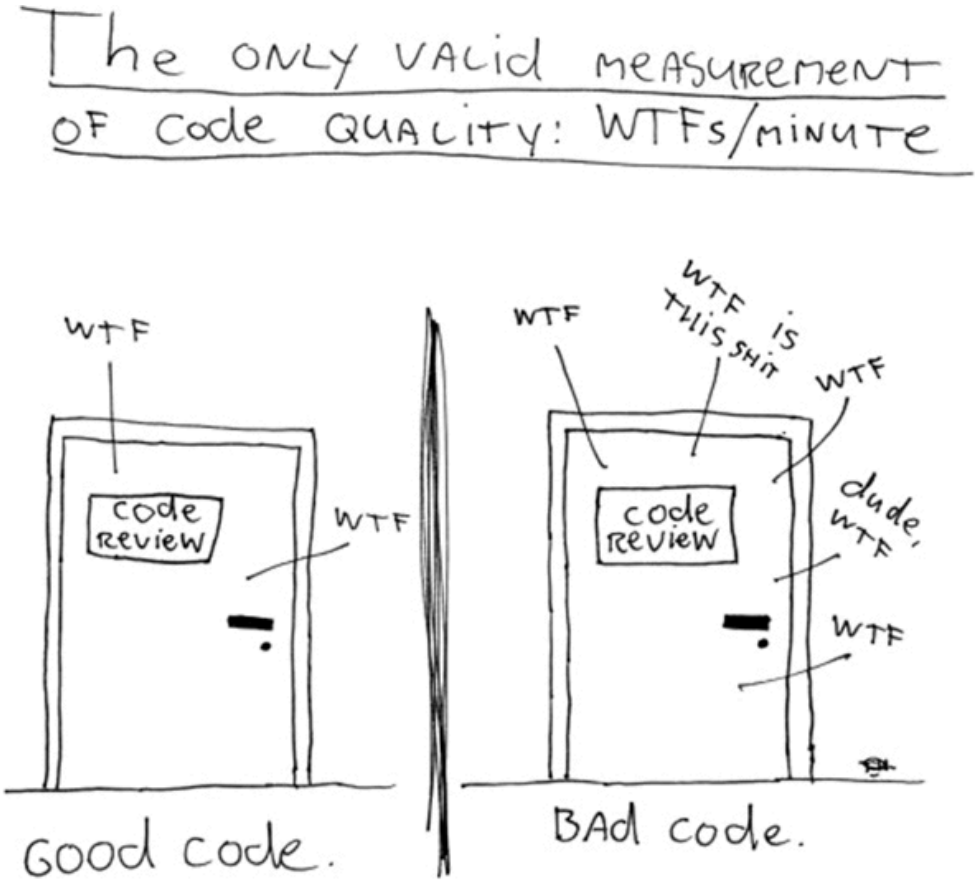
Introduction (from Long Ago)

This is the Introduction to the first edition and has been edited and abridged.



Reproduced with the kind permission of Thom Holwerda (www.osnews.com/story/19266/WTFs_m).

Look at the drawing above. Which door represents your code? Which door represents your team or your company? Why are we in that room? Is this just a normal code review or have we found a stream of horrible problems shortly after going live?

Are we debugging in a panic, poring over code that we thought worked? Are customers leaving in droves and managers breathing down our necks? How can we make sure we wind up behind the right door when the going gets tough? The answer is: craftsmanship.

There are two parts to learning craftsmanship: knowledge and work. You must gain the knowledge of principles, patterns, practices, and heuristics that a craftsman knows, and you must also grind that knowledge into your fingers, eyes, and gut by working hard and practicing.

I can teach you the physics of riding a bicycle. Indeed, the classical mathematics is relatively straightforward. Gravity, friction, angular momentum, center of mass, and so forth can be demonstrated with less than a page full of equations. Given those formulae, I could prove to you that bicycle riding is practical and give you all the knowledge you needed to make it work.

And you'd still fall down the first time you climbed on that bike.

Coding is no different. We could write down all the "feel good" principles of clean code and then trust you to do the work (in other words, let you fall down when you get on the bike), but then what kind of teachers would that make us, and what kind of student would that make you?

No. That's not the way this book is going to work.

Learning to write clean code is hard work. It requires more than just the knowledge of principles and patterns. You must *sweat* over it. You must practice it yourself, and watch yourself fail. You must watch others practice it and fail. You must see them stumble and retrace their steps. You must see them agonize over decisions and see the price they pay for making those decisions the wrong way.

Be prepared to work hard while reading this book. This is not a "feel good" book that you can read on an airplane and finish before you land. This book will make you work, *and work hard*. What kind of work will you be doing? You'll be reading code—lots of code. And you will be challenged to think about what's right about that code and what's wrong with it. You'll be asked to follow along as we (the authors) take modules apart and put them back together again. This will take time and effort; but we think it will be worth it.