Making Quality Software

at End Point

Jon Jensen, 15 May 2019

How good is good enough?

How “nice” should we make software?

How robust does it need to be?

How quickly do we have to do the work?

How much time can we spend on testing?

Can we have someone review our code?

Who pays?

**Everything we do
takes time
and thus adds cost.**

Who pays?

Things we don’t do initially often add time later and thus also add cost.

Cost = money?

Client’s time has value.

Client’s reputation with their customers has value.

Software testing:
A review

Whitebox testing

takes into account the internal mechanism of a system.

Also called structural testing and glass box testing.

Unit testing

is testing of an individual code unit (function, method, etc.) or group of related units.

Integration testing

is testing a group of code components in combination.

Blackbox testing

ignores the internal mechanism of the system.

Focuses on the output generated against any input and execution of the system.

Functional testing

ensures that the specified functionality required in the system requirements works.

System testing

puts the software in different environments (operating systems, browsers, screen sizes) to ensure it works everywhere expected.

Usability (UX) testing

shows how easily users accomplish tasks, recover from mistakes, understand the system.

Stress testing

sees how the system behaves under bad conditions, beyond limits of the specs, using bad data, with insufficient and unreliable resources.

Performance testing

assesses the speed and effectiveness of the system to ensure it generates results within expected time.

Cross-cutting test types

These overlap some of the previous ones.

Smoke testing

checks basic functionality of the application, to assure us that major features work as expected.

Quick to execute.

End-to-end testing

exercises the whole flow from start (new user, old user login) to finish (ecommerce order, document search).

Acceptance testing

is any testing done by the customer to accept the work and consider it done.

Regression testing

is done after any modification to ensure that the changed area works and that there is no collateral damage to other areas.

Beta testing

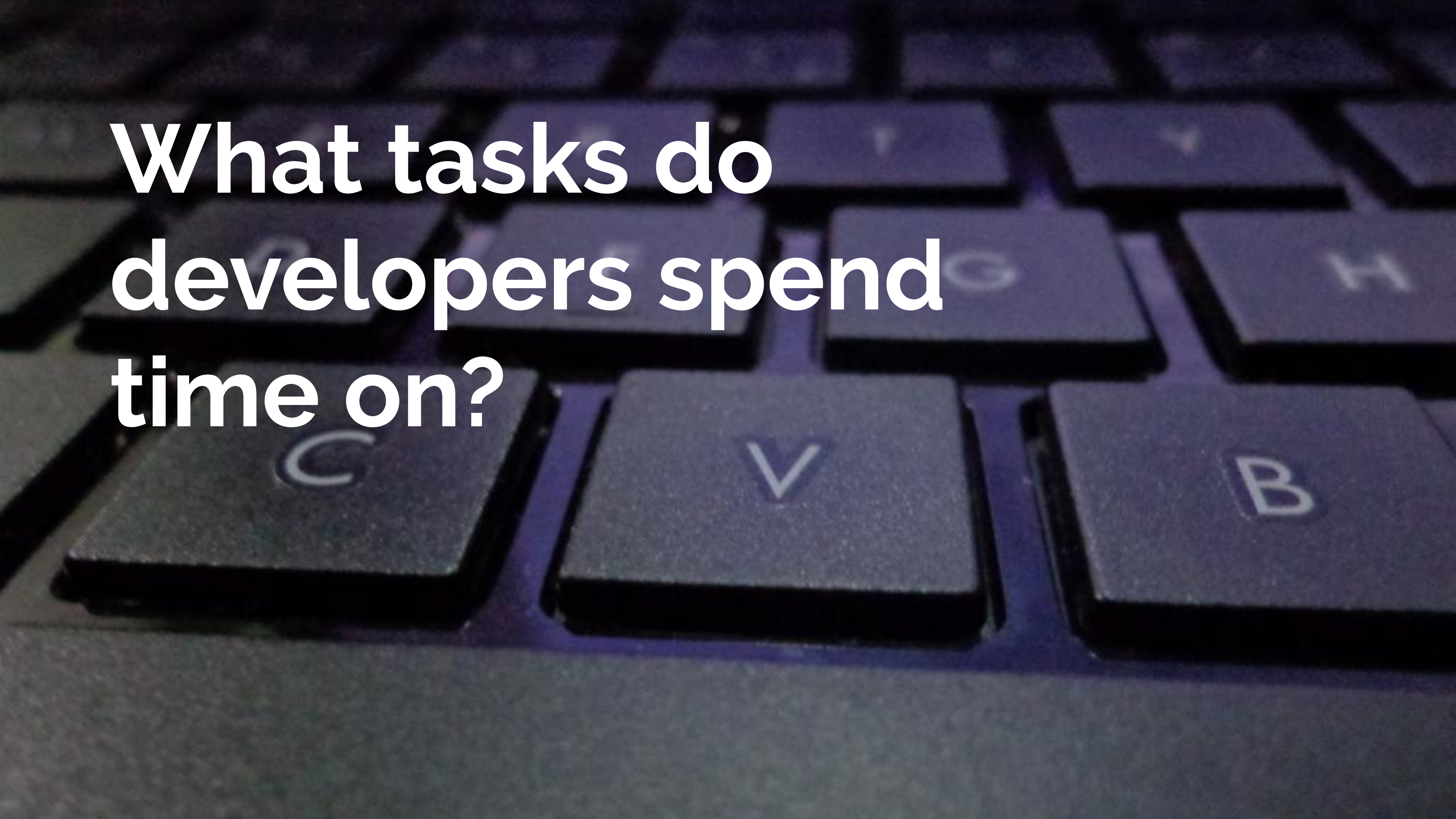
is done by a subset of end users on a public release of the software to find unexpected problems before they will affect all users.

Manual vs. automated

Some of the types of tests mentioned can be automated.

When they can, they should be!

But some manual testing will always be needed.



What tasks do developers spend time on?

Project management

Juggling scheduled projects, deadlines, surprise work

Estimates and planning

Considering interactions between projects and systems

Gathering requirements

Client’s idea may be vague

May be specific but difficult or expensive or risky

Looking for effects on other systems

Writing code

Greenfield programs

New features on old codebases

Maintenance: bugfixes, adapting to environment changes

Manual QA, debugging

Try it out, fix what's broken

In how many environments?

For how long before passing it to the client?

Code review

By yourself

By others

Pair programming or review later?

Before or after trouble?

Documenting

Work done and time spent: timesheets, reports

Documentation for users, developers, hosting

Git commit messages, production rollout logs

Production support

Monitoring and alerting

Performance testing, security audits

Disaster recovery, business continuity

Let’s take a poll

http://www.polljunkie.com/poll/sokrrx/making-quality-software

Doing same tasks in different order

Preventative or reactive?

Is one way better?

Possible to predict everything?

Custom development

always involves uncertainty.

It always depends

at End Point:

on the client, budget, timeline, problem space, team, your ambition and flexibility.

Together we make our culture.

Try something new:
Faster or more deliberative,
Simpler or more complicated,
Different collaboration.

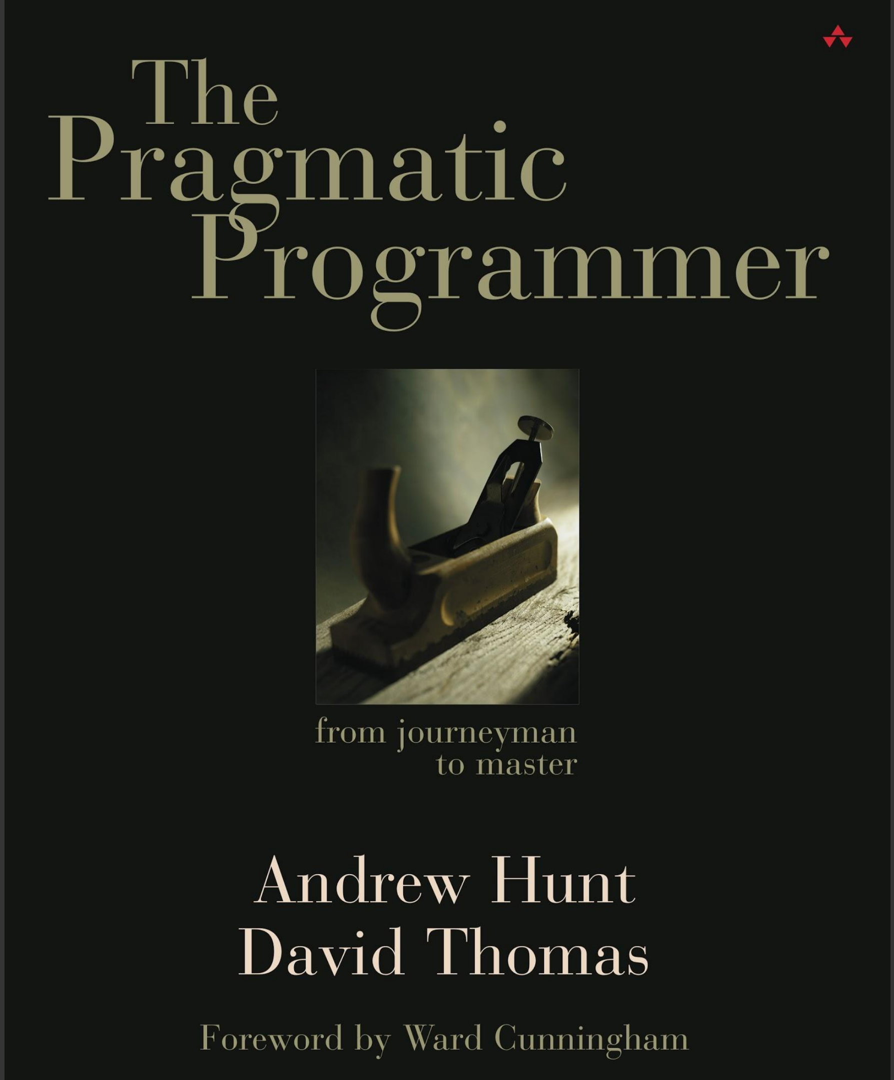
References

On testing:

https://www.codeproject.com/tips/351122/what-is-software-testing-what-are-the-different-ty

https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing

And some favorite books …

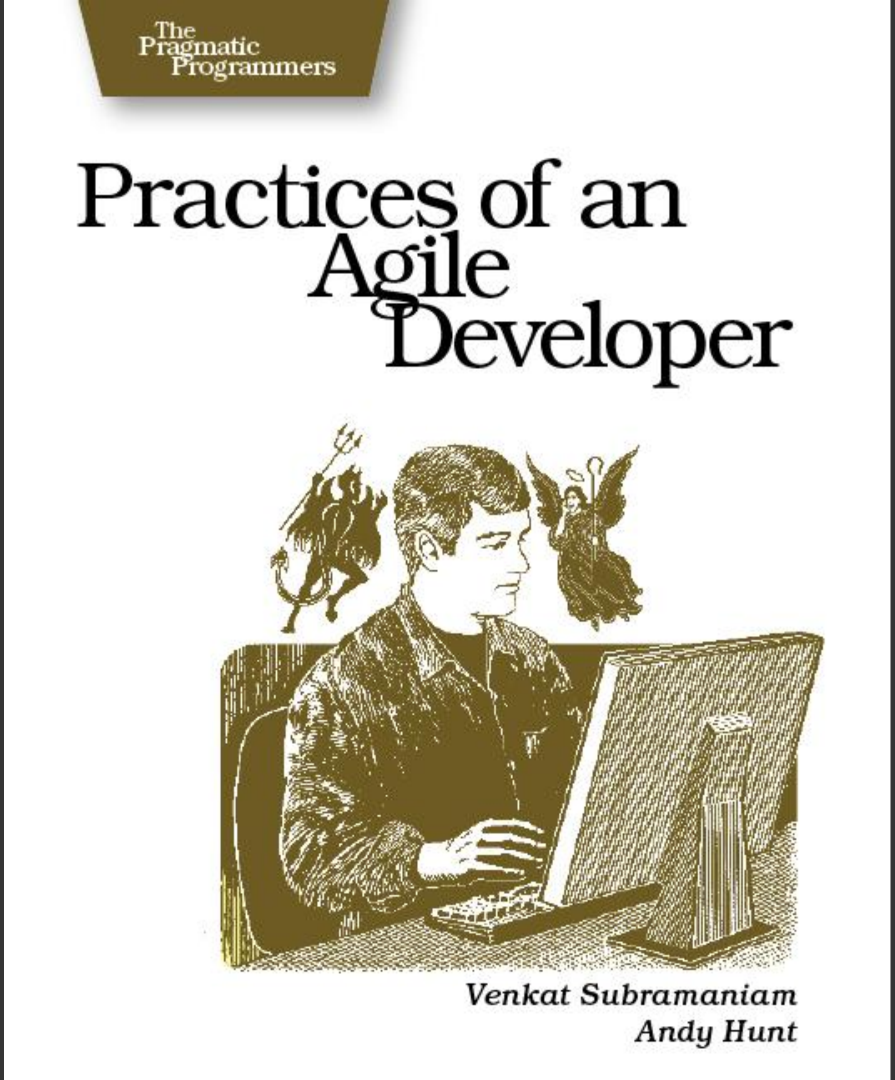


The Pragmatic Programmer

from journeyman to master

Andrew Hunt
David Thomas

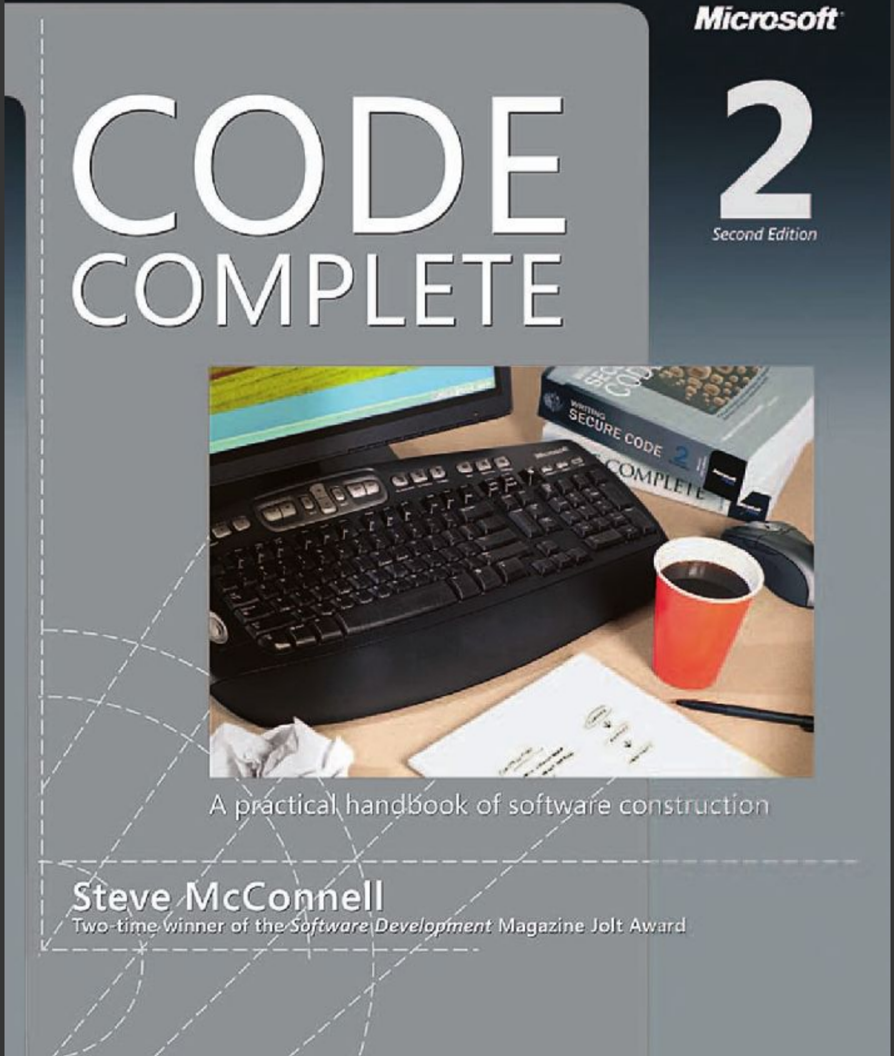
Foreword by Ward Cunningham



The Pragmatic Programmers

Practices of an Agile Developer

Venkat Subramaniam
Andy Hunt


Microsoft

CODE COMPLETE

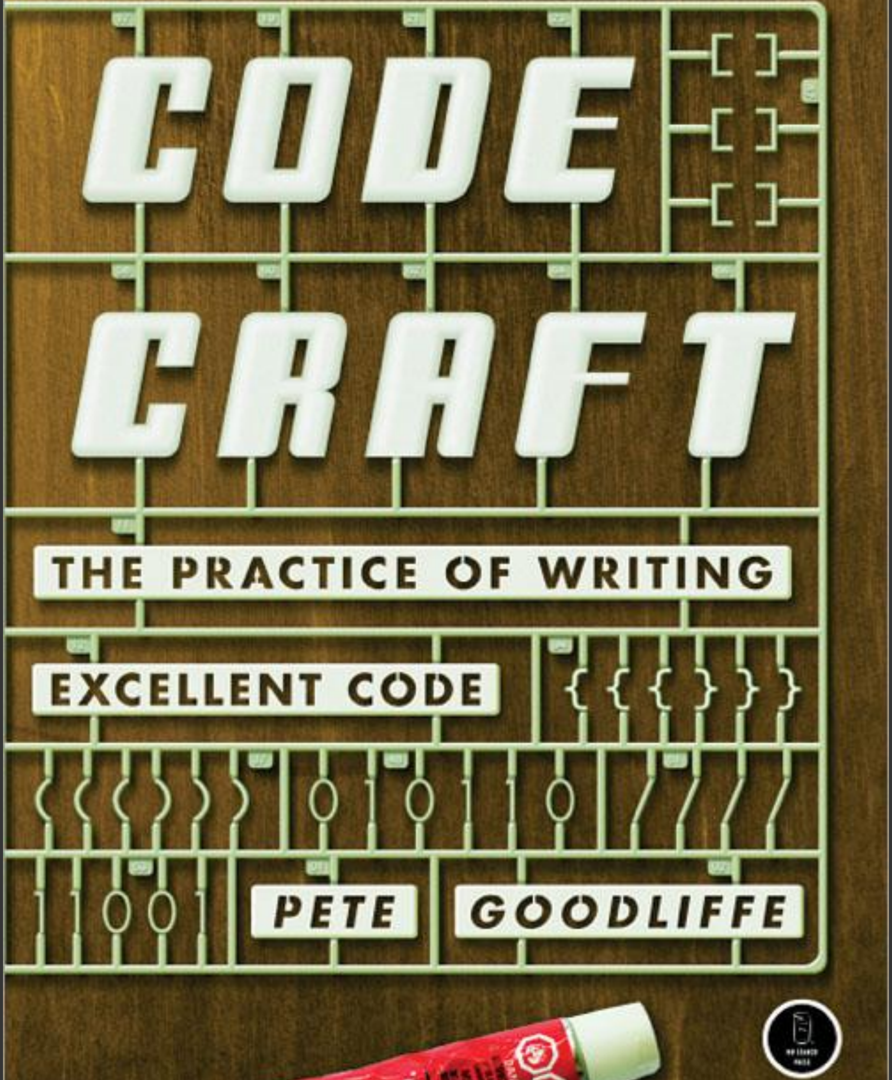
2

Second Edition

A practical handbook of software construction

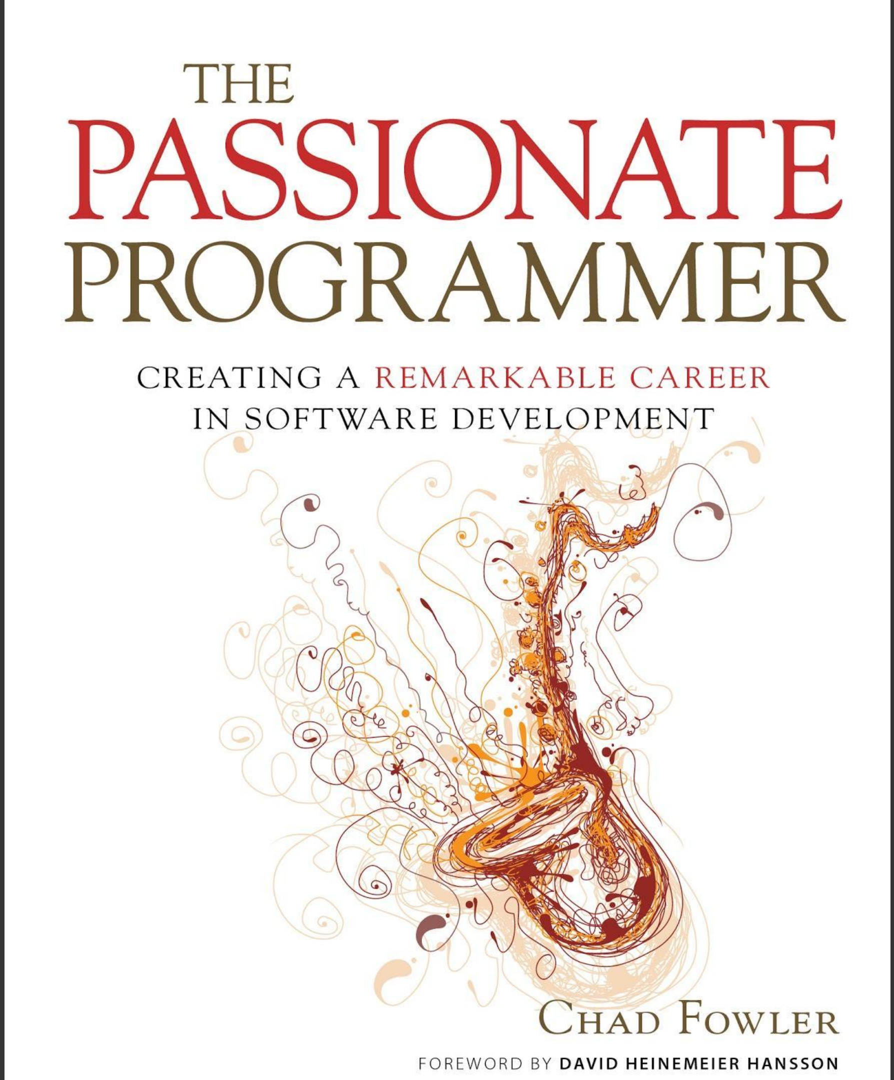
Steve McConnell

Two-time winner of the *Software Development* Magazine Jolt Award


CODE CRAFT

THE PRACTICE OF WRITING EXCELLENT CODE

PETE GOODLIFFE


THE PASSIONATE PROGRAMMER

CREATING A REMARKABLE CAREER IN SOFTWARE DEVELOPMENT

CHAD FOWLER

FOREWORD BY DAVID HEINEMEIER HANSSON

What books, blogs, etc. do you recommend?

Poll results

http://www.polljunkie.com/poll/rinpoq/making-quality-software/view